MINISTRY OF EDUCATION AND TRAINING

Hà Nội, January 2015

Crazy Racing

Capstone Project

6S-Team	
Supervisor	Trần Bình Dương
Group members	Đinh Ngọc Sơn Đinh Tuấn Mạnh Lê Việt Anh Trần Quốc Duy Tôn Thất Hoàng Triều Nguyễn Cao Cường
Capstone Project Code	CR1

Chapter 1 - Introduction

This document presents an overview of the project include: the reality of the game market, the purpose of this project, the project participants, some same existing game, proposal, applied technology and some core function…

I. Project Information

CR1 is 3D RPG game, Player is in role as a driver running fast in the freeway that have to find out way to evade the obstacles. CR1 create by Unity3D tool and C# .Net.

- Project Name	: Crazy Racing.
- Project Code	: CR1
- Product Type	: Mobile Application
- Timeline	: From Jan 2015 to Apr 2015

II. Purposes

This project is registered and implemented as a capstone project for all team members, apply knowledge that studied at FPT University to complete well. Furthermore, we want to create a complete product for going live. Complete this project will be a good change for all team, members can learn some new technologies: Unity3D, develop in iOS & Android platform… and some others soft skills: working in group, Communication skill,…etc

III. People

Supervisor:

Full name	Phone	E-mail	Role in Group
Trần Bình Dương	0936168165	duongtb@fpt.edu.vn	Supervisor

Team member:

No	Student code	Full name	Phone	E-mail	Role in Group
1	01743	Đinh Ngọc Sơn	01649589787	sondn01743@fpt.edu.vn	PM

2	SE60643	Trần Quốc Duy	0906577104	duytqse60643@fpt.edu.vn	Developer
3	60114	Nguyễn Cao Cường	0985 456 222	cuongnc60114@fpt.edu.vn	Developer
4	01513	Lê Việt Anh	01658642297	anhlv01513@fpt.edu.vn	Tester
5	01938	Đinh Tuấn Mạnh	01652583353	manhdt01938@fpt.edu.vn	Developer
6	SE60603	Tôn Thất Hoàng Triều	01676578910	trieutthse60603@fpt.edu.vn	Developer

IV. Background

Nowadays, game is one of important fields in entertainment area. Moreover, smartphone is growing up promptly, almost of people also have a personal smartphone. Because of that, game mobile development is very necessary. In fact, there were a lot of successful products that had gotten high profit such as: Flappy bird, angry bird, ninja fruits…

Indeed, Viet Nam's game development field is still in growing process and not really have games that satisfy the market. Consequently, we've decided to develop a game for people's relaxing in their free time on iOS and Android.

V. Overview of existing product:

Some mobile games are already released on market:

1. Racing moto:

Figure 1.1: Racing Moto

- **Highlights:**

The road in this game contains two partitioning line dividing it into three lanes which is used by the vehicles to switch their sides in the game. All you have to do is to drive your vehicle on one of the two lines where there is no obstacle. But, you need to be careful because the vehicles coming into your way often change their direction which can crash you down if you hit any of them while in full speed. So, again you will need to have a good hold on the controls of the game to keep your vehicle on the right track without any accident or chance of crashing**.** The store of the Racing moto offers you two more amazing bike to ride on them. These two bikes are slimmer than the default bike and help you to escape even the narrower spaces between the two vehicles and hence increasing your chancing to creating a big score. The bikes can be unlocked by reaching a score of 30000 and 70000 in a single game of more than one game.

The traffic gets denser as we cross the score of 50,000. You will also notice that it is a great fun to drive through heavy traffic and after playing the game for about 10-15 days, you will feel like an expert.

- **Limitation**:

No in-app purchase, no multiplayer, no Facebook share.

2. Bike racing 2014:

Figure 1.2 The Racing 2014

- **Highlights:**

  A new bike drag racing game with exciting levels and amazing HD graphics with smooth physics control. If you are crazy about racing game and crazy about bikes, then try this game to enjoy. Buy & upgrade new sports bikes to boost your power. Shift your gear and race in different locations & cities.

- **Limitation:**

  Players can use bikes like the Super Bike and Ultimate Bike in competitive mode. These bikes require no skill, and completely takes away the fun for players. I totally get being able to use them in single player, but for competitive multiplayer it's just ridiculous. Additionally, I'm constantly being bombarded with please to buy new bikes from a game that I already purchased. There's a free version for a reason, and these ads should stay in the free version and be taken out of the pad version.

VI. Proposal

The idea about motor racing game is not new, even though there are many product with the same format. But we expect that our game will bring to players satiation by integrating multiple features.

Crazy Racing will provide to players the following features:

- Interesting game-play.
- In-app purchase providing.

- Facebook sharing.
- Share achievement by game center.

VII. Product

The main product of this project is a game as mentioned above. This is a running game with the following function:

- View, change game settings
  - View maps (choose – unlock maps)
  - View cars (choose – unlock cars)
  - Choose music
  - Pause game
- Drive car
  - Control car (Turn – Brake – Use Boost Energy)
  - Pick up items
- View information
  - View Achievements
  - View Pick up coins
  - View Driven distance
  - View high scores
- Share facebook
- Purchase game coins
- Play with the other by Multiplayer

VIII. Technology

To develop this project, some technical need to use:

- **Unity 3D**: C# .Net

Figure 1.3: Unity3D

Unity is a game development ecosystem: a powerful rendering engine fully integrated with a complete set of intuitive tools and rapid workflows to create interactive 3D and 2D content; easy multiplatform publishing; thousands of quality, ready-made assets in the Asset Store and a knowledge-sharing community.

For independent developers and studios, Unity's democratizing ecosystem smashes the time and cost barriers to creating uniquely beautiful games. They are using Unity to build a livelihood doing what they love: creating games that hook and delight players on any platform.

Chapter 2 - Project management plan

I. Problem Definition

1. Name of this Capstone Project

The official and formal Capstone project name is Crazy Racing – a mobile game for iOS and Android operating system. This game provides an interesting and funny challenging for user's relaxing time.

2. Problem abstract

Nowadays, game is one of important fields in entertainment area. Moreover, smartphone is growing up promptly, almost of people also have a personal smartphone. Because of that, game mobile development is very necessary. In fact, there were a lot of successful products that had gotten high profit such as: Flappy bird, angry bird, ninja fruits.

Indeed, VN's game development field is still in growing process and not really have games that satisfy the market. Consequently, we've decided to develop a game for people's relaxing in their free time on iOS and Android.

There are some other products providing gameplay and graphic but all are not good enough. The purpose of this project is to create a product that overcomes all current products.

3. Project overview

3.1 The current system

After researched from existing games, almost from Play store, we found out that have many games those similar to our game-style. Racing car, motorbike, running.

3.2 The Proposed System

From all the limitation of existing applications, we are going to produce an application that could be run in two popular platforms iOS & android.

Basic Functions:

- Racing: control the car by sensor, multi touch.
- Let users get power-up items, coins, special abilities.
- Share achievement by game center.
- Multiplayer connect two players via internet connection.
- In-app purchase to buy Items.
- Friendly graphic.

3.3 Boundaries of the System

The Software under development of this Capstone project will include the complete of iOS and Android mobile game with all functions defined in the requirements.

3.4 Development Environment

3.4.1 Hardware environment

- Personal computers for developing with the minimum configuration: 2 GB of RAM, 50 GB of hard disk, Pentium dual core 2.0 GHz.
- iPhone, Android mobile.

3.4.2 Software environment

- Operating system: Mac OS, Windows
- IDE : XCode , Unity3D, 3D Max, Blender.
- Source control: SVN.

II. Software Organization

1. Process Model

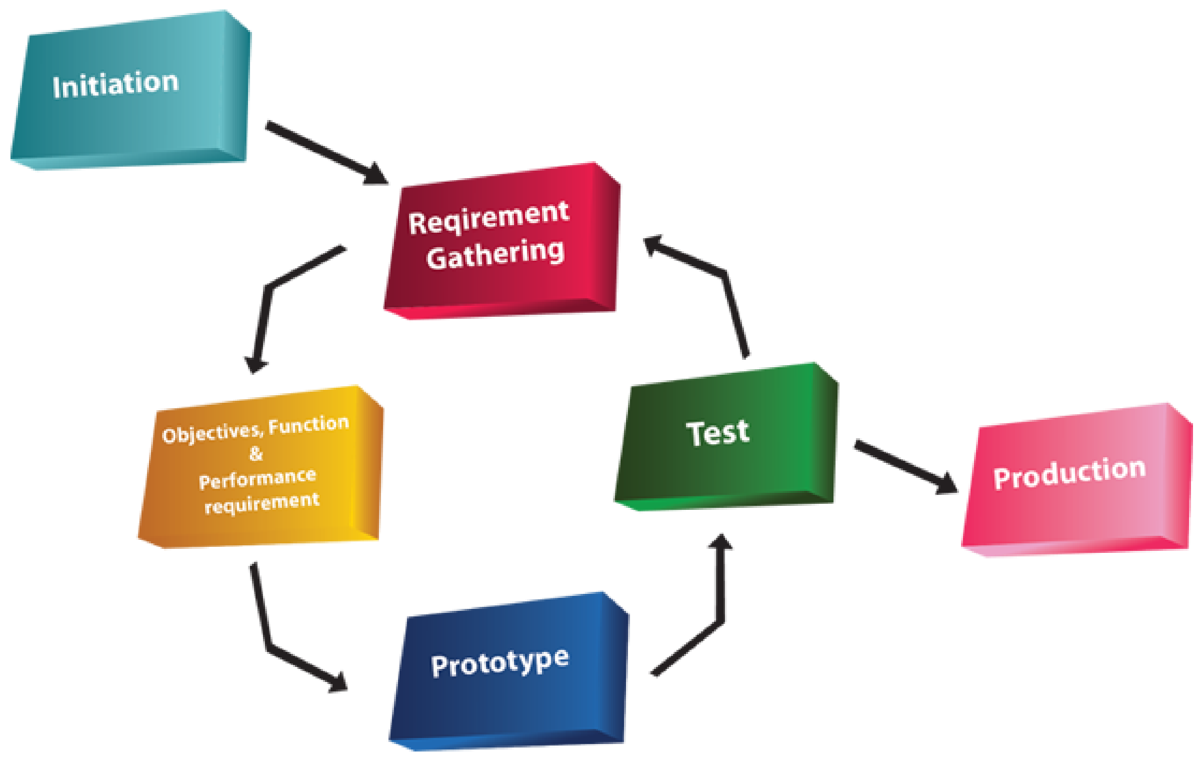


Figure 2.1: Prototype model

The Prototyping Model is a systems development method in which a prototype (prototype is an early approximation of a final system or product) is built, tested, and then reworked as necessary until an acceptable prototype is finally achieved from which the complete system or product can now be developed. This model works best in scenarios where not all of the project requirements are known in detail ahead of time. It is an iterative, trial-and-error process that takes place between the developers and the users.

Pros	**Cons**
Increased user involvement in the product even before implementation	Risk of insufficient requirement analysis owing to too much dependency on prototype
Since a working model of the system is	Users may get confused in the prototypes

displayed, the users get a better understanding of the system being developed.	and actual systems.
Reduces time and cost as the defects can be detected much earlier.	Practically, this methodology may increase the complexity of the system as scope of the system may expand beyond original plans.
Quicker user feedback is available leading to better solutions.	Developers may try to reuse the existing prototypes to build the actual system, even when its not technically feasible
Missing functionality can be identified easily	The effort invested in building prototypes may be too much if not monitored properly
Confusing or difficult functions can be identified	Risk of insufficient requirement analysis owing to too much dependency on prototype

2. Roles and Responsibilities

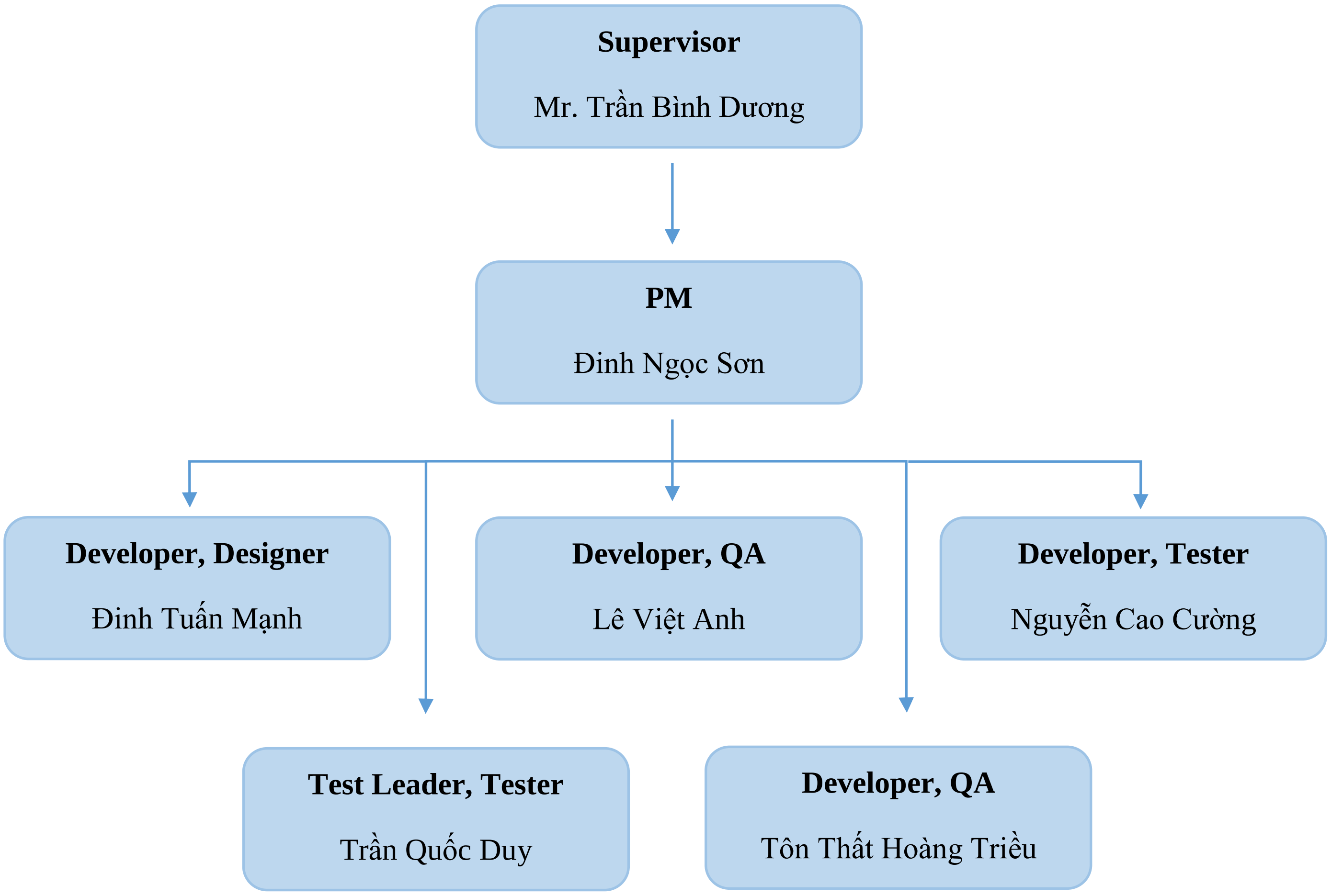


Figure 2.1: Roles & responsibility

No	Name	Role	Responsibilities
1	**Đinh Ngọc Sơn**	PM	- Managing process - Clarifying requirements - Training - Coding
2	**Đinh Tuấn Mạnh**	Developer Designer	- GUI design - Coding - Managing configuration

			- Testing
3	**Lê Việt Anh**	Developer QA	- Coding - Testing - Quality control - GUI design
4	**Nguyễn Cao Cường**	Developer Tester	- Testing - Coding - Verifying requirement - Creating system test case
5	**Trần Quốc Duy**	Test Leader Tester	- Creating test case - Creating test plan - Clarifying requirements - Coding
6	**Tôn Thất Hoàng Triều**	Developer QA	- Quality control - Coding - Testing - Managing document

3. Tools and techniques

No	**Tools & Techniques**	**Description**
1	SmartSVN	For managing documents, source codes...
2	Skype	For Communication
3	Microsoft Project 2013	For making project plan
4	Microsoft world 2013	For documents
5	Microsoft excel 2013	For report, making test cases
6	Software moduler	For creating UML items
7	Unity3D	IDE for coding
8	Visual Studio	IDE for C# .Net coding
9	xCode	IDE for build iOS
10	Blender, 3D Max, PS	IDE for design graphic

III. Project Management Plan

1. Project Milestones and Deliverable

No	Date	Milestone name	Comment
1	Febuary 5, 2015	Prototype 1 (0.1)	This prototype prodives basic control car funtions such as turn left and right and camera movement. Images is simple, they are just cubic or sphere
2	Febuary 14, 2015	Prototype 2 (0.2)	In this prototype, the car can hit the power up item and traffic car.
3	March 5, 2015	Prototype 3 (0.3)	In this prototype, the game have new design and 2 new functions: highscore, share achivement.
4	March 20, 2015	Prototype 4 (0.4)	In this prototype, the game show first look about multiplayer. How to to client connect to gether, how to sovle synchronous problem.
5	April 1, 2015	Prototype 5 (0.5)	In this protype, fix some bugs and view that is collected by testing process.
6	April 10, 2015	Beta 1 (0.6)	Full functions with final design.
7	April 20, 2015	Release 1 (0.7)	This version is on market.

2. Tasks

2.1 Create Software Requirement Specification

- Description: Create software requirement specification.
- Output: SRS document.
- Deliverables: January 14, 2015
- Resources needed: 6 people for 2 weeks.
- Dependencies and Constraints: None.
- Risk: lack of game analysis skill, requirements may not clear.

2.2 Create Software Design Description

- Description: Create software design description
- Output: SDD document
- Deliverables: January 25, 2015
- Resources needed: 6 people for 10 days
- Dependencies and Constraints: SRS document and supervisor
- Risk: Weak software structure, hard to predict problem of requirement

2.3 Graphic Design

- Description: Create graphics of game

- Output: all image that used in game 3d extension file
- Deliverables: April 10, 2015
- Resources needed: 2 people for 2 months
- Dependencies and Constraints: Other game graphics, SRS, SDD
- Risk: Image is not good as expected

2.4 Coding

- Description: Create code that make game play
- Output: Source code
- Deliverables: April 15, 2015
- Resources needed: 2 people for 2 months
- Dependencies and Constraints: SDD
- Risk: Lack of game coding skill, lack experience with Unity

2.5 Create Test Plan

- Description: Schedule of testing
- Output: Test plan document
- Deliverables: February 15, 2015
- Resources needed: 1 people for 3 days
- Dependencies and Constraints: SDD
- Risk: Under or overestimate works

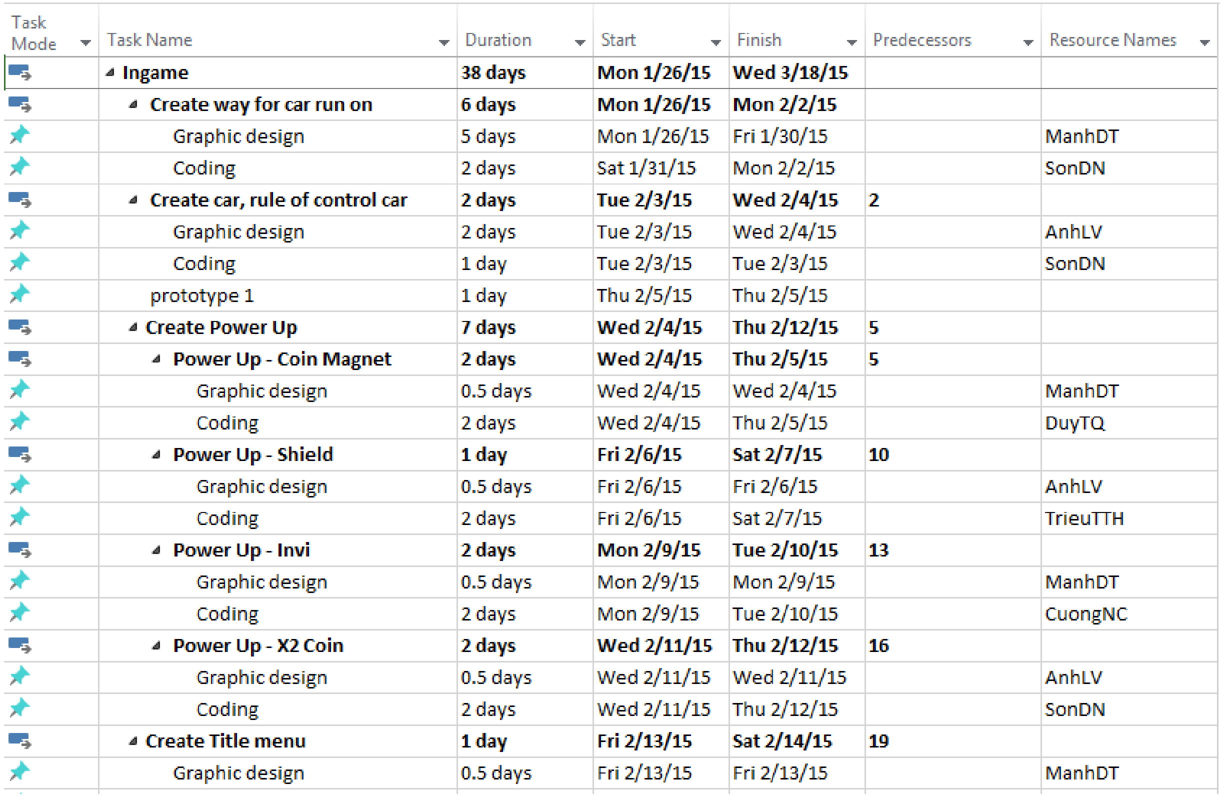
2.6 Testing

- Description: Test game
- Output: Testing report
- Deliverables: 20 April, 2015
- Resources needed: 2 people for 2 months
- Dependencies and Constraints: Source code, SDD
- Risk: Lack of experience, Japanese

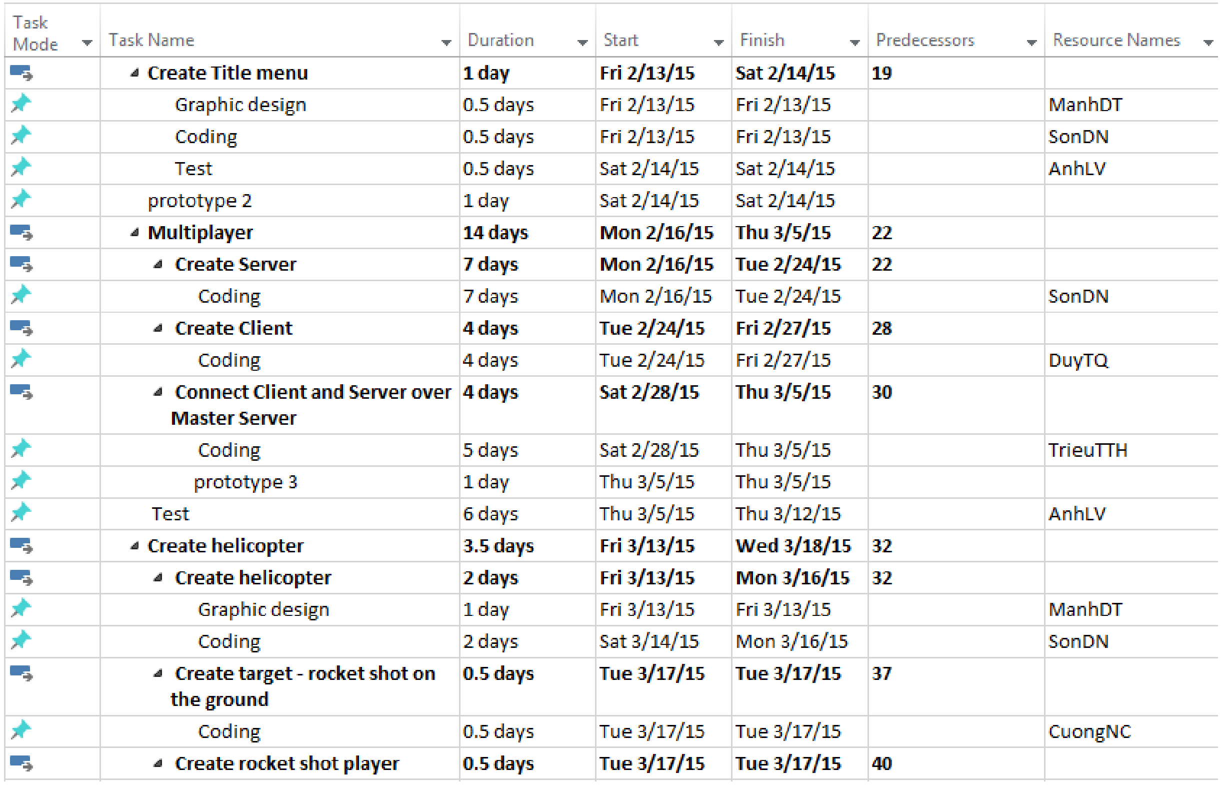
2.7 Create User Manual

- Description: Create user manual
- Output: User guide
- Deliverables: 20 April, 2015
- Resources needed: 1 person for 3 days.
- Dependencies and Constraints: SDD and release product
- Risk: Japanese

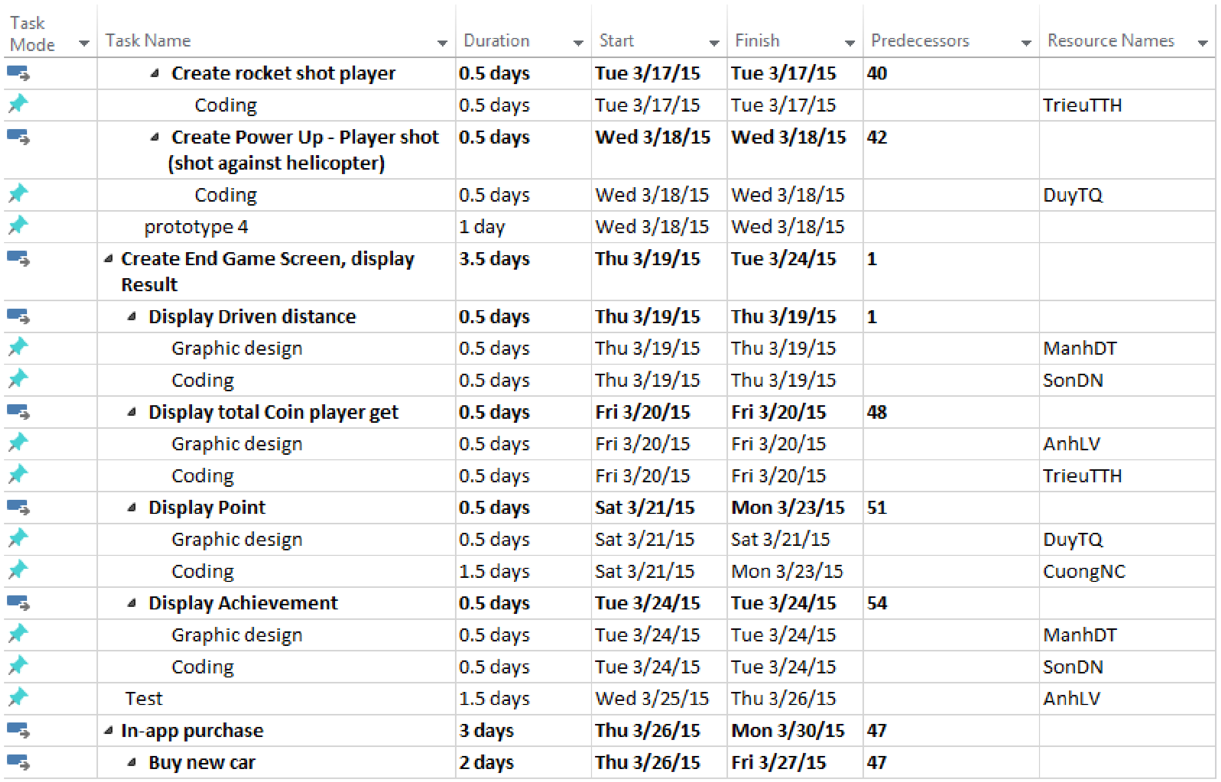
3. Assignment and Timetable:



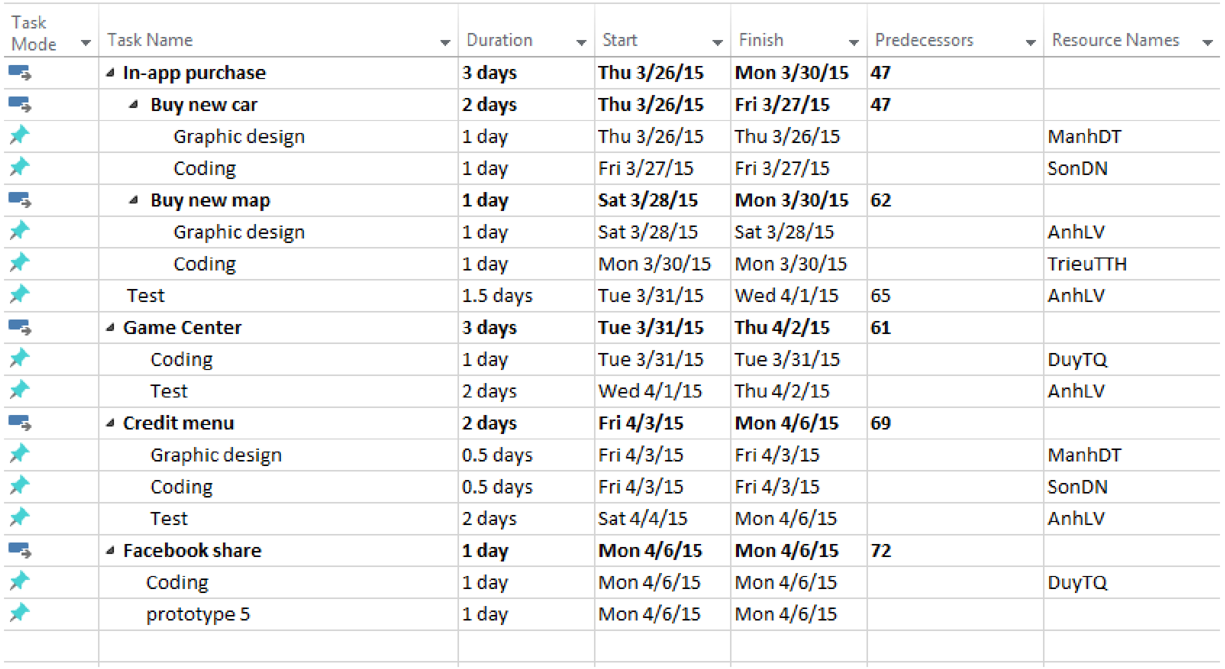
Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names
➡	Ingame	38 days	Mon 1/26/15	Wed 3/18/15		
➡	Create way for car run on	6 days	Mon 1/26/15	Mon 2/2/15		
📌	Graphic design	5 days	Mon 1/26/15	Fri 1/30/15		ManhDT
📌	Coding	2 days	Sat 1/31/15	Mon 2/2/15		SonDN
➡	Create car, rule of control car	2 days	Tue 2/3/15	Wed 2/4/15	2	
📌	Graphic design	2 days	Tue 2/3/15	Wed 2/4/15		AnhLV
📌	Coding	1 day	Tue 2/3/15	Tue 2/3/15		SonDN
📌	prototype 1	1 day	Thu 2/5/15	Thu 2/5/15		
➡	Create Power Up	7 days	Wed 2/4/15	Thu 2/12/15	5	
➡	Power Up - Coin Magnet	2 days	Wed 2/4/15	Thu 2/5/15	5	
📌	Graphic design	0.5 days	Wed 2/4/15	Wed 2/4/15		ManhDT
📌	Coding	2 days	Wed 2/4/15	Thu 2/5/15		DuyTQ
➡	Power Up - Shield	1 day	Fri 2/6/15	Sat 2/7/15	10	
📌	Graphic design	0.5 days	Fri 2/6/15	Fri 2/6/15		AnhLV
📌	Coding	2 days	Fri 2/6/15	Sat 2/7/15		TrieuTTH
➡	Power Up - Invi	2 days	Mon 2/9/15	Tue 2/10/15	13	
📌	Graphic design	0.5 days	Mon 2/9/15	Mon 2/9/15		ManhDT
📌	Coding	2 days	Mon 2/9/15	Tue 2/10/15		CuongNC
➡	Power Up - X2 Coin	2 days	Wed 2/11/15	Thu 2/12/15	16	
📌	Graphic design	0.5 days	Wed 2/11/15	Wed 2/11/15		AnhLV
📌	Coding	2 days	Wed 2/11/15	Thu 2/12/15		SonDN
➡	Create Title menu	1 day	Fri 2/13/15	Sat 2/14/15	19	
📌	Graphic design	0.5 days	Fri 2/13/15	Fri 2/13/15		ManhDT



Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names
➡	Create Title menu	1 day	Fri 2/13/15	Sat 2/14/15	19	
📌	Graphic design	0.5 days	Fri 2/13/15	Fri 2/13/15		ManhDT
📌	Coding	0.5 days	Fri 2/13/15	Fri 2/13/15		SonDN
📌	Test	0.5 days	Sat 2/14/15	Sat 2/14/15		AnhLV
📌	prototype 2	1 day	Sat 2/14/15	Sat 2/14/15		
➡	Multiplayer	14 days	Mon 2/16/15	Thu 3/5/15	22	
➡	Create Server	7 days	Mon 2/16/15	Tue 2/24/15	22	
📌	Coding	7 days	Mon 2/16/15	Tue 2/24/15		SonDN
➡	Create Client	4 days	Tue 2/24/15	Fri 2/27/15	28	
📌	Coding	4 days	Tue 2/24/15	Fri 2/27/15		DuyTQ
➡	Connect Client and Server over Master Server	4 days	Sat 2/28/15	Thu 3/5/15	30	
📌	Coding	5 days	Sat 2/28/15	Thu 3/5/15		TrieuTTH
📌	prototype 3	1 day	Thu 3/5/15	Thu 3/5/15		
📌	Test	6 days	Thu 3/5/15	Thu 3/12/15		AnhLV
➡	Create helicopter	3.5 days	Fri 3/13/15	Wed 3/18/15	32	
➡	Create helicopter	2 days	Fri 3/13/15	Mon 3/16/15	32	
📌	Graphic design	1 day	Fri 3/13/15	Fri 3/13/15		ManhDT
📌	Coding	2 days	Sat 3/14/15	Mon 3/16/15		SonDN
➡	Create target - rocket shot on the ground	0.5 days	Tue 3/17/15	Tue 3/17/15	37	
📌	Coding	0.5 days	Tue 3/17/15	Tue 3/17/15		CuongNC
➡	Create rocket shot player	0.5 days	Tue 3/17/15	Tue 3/17/15	40	



Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names
➡	**Create rocket shot player**	**0.5 days**	**Tue 3/17/15**	**Tue 3/17/15**	**40**	
📌	Coding	0.5 days	Tue 3/17/15	Tue 3/17/15		TrieuTTH
➡	**Create Power Up - Player shot (shot against helicopter)**	**0.5 days**	**Wed 3/18/15**	**Wed 3/18/15**	**42**	
📌	Coding	0.5 days	Wed 3/18/15	Wed 3/18/15		DuyTQ
📌	prototype 4	1 day	Wed 3/18/15	Wed 3/18/15		
➡	**Create End Game Screen, display Result**	**3.5 days**	**Thu 3/19/15**	**Tue 3/24/15**	**1**	
➡	**Display Driven distance**	**0.5 days**	**Thu 3/19/15**	**Thu 3/19/15**	**1**	
📌	Graphic design	0.5 days	Thu 3/19/15	Thu 3/19/15		ManhDT
📌	Coding	0.5 days	Thu 3/19/15	Thu 3/19/15		SonDN
➡	**Display total Coin player get**	**0.5 days**	**Fri 3/20/15**	**Fri 3/20/15**	**48**	
📌	Graphic design	0.5 days	Fri 3/20/15	Fri 3/20/15		AnhLV
📌	Coding	0.5 days	Fri 3/20/15	Fri 3/20/15		TrieuTTH
➡	**Display Point**	**0.5 days**	**Sat 3/21/15**	**Mon 3/23/15**	**51**	
📌	Graphic design	0.5 days	Sat 3/21/15	Sat 3/21/15		DuyTQ
📌	Coding	1.5 days	Sat 3/21/15	Mon 3/23/15		CuongNC
➡	**Display Achievement**	**0.5 days**	**Tue 3/24/15**	**Tue 3/24/15**	**54**	
📌	Graphic design	0.5 days	Tue 3/24/15	Tue 3/24/15		ManhDT
📌	Coding	0.5 days	Tue 3/24/15	Tue 3/24/15		SonDN
📌	Test	1.5 days	Wed 3/25/15	Thu 3/26/15		AnhLV
➡	**In-app purchase**	**3 days**	**Thu 3/26/15**	**Mon 3/30/15**	**47**	
➡	**Buy new car**	**2 days**	**Thu 3/26/15**	**Fri 3/27/15**	**47**	



Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names
➡	**In-app purchase**	**3 days**	**Thu 3/26/15**	**Mon 3/30/15**	**47**	
➡	**Buy new car**	**2 days**	**Thu 3/26/15**	**Fri 3/27/15**	**47**	
📌	Graphic design	1 day	Thu 3/26/15	Thu 3/26/15		ManhDT
📌	Coding	1 day	Fri 3/27/15	Fri 3/27/15		SonDN
➡	**Buy new map**	**1 day**	**Sat 3/28/15**	**Mon 3/30/15**	**62**	
📌	Graphic design	1 day	Sat 3/28/15	Sat 3/28/15		AnhLV
📌	Coding	1 day	Mon 3/30/15	Mon 3/30/15		TrieuTTH
📌	Test	1.5 days	Tue 3/31/15	Wed 4/1/15	65	AnhLV
➡	**Game Center**	**3 days**	**Tue 3/31/15**	**Thu 4/2/15**	**61**	
📌	Coding	1 day	Tue 3/31/15	Tue 3/31/15		DuyTQ
📌	Test	2 days	Wed 4/1/15	Thu 4/2/15		AnhLV
➡	**Credit menu**	**2 days**	**Fri 4/3/15**	**Mon 4/6/15**	**69**	
📌	Graphic design	0.5 days	Fri 4/3/15	Fri 4/3/15		ManhDT
📌	Coding	0.5 days	Fri 4/3/15	Fri 4/3/15		SonDN
📌	Test	2 days	Sat 4/4/15	Mon 4/6/15		AnhLV
➡	**Facebook share**	**1 day**	**Mon 4/6/15**	**Mon 4/6/15**	**72**	
📌	Coding	1 day	Mon 4/6/15	Mon 4/6/15		DuyTQ
📌	prototype 5	1 day	Mon 4/6/15	Mon 4/6/15		

Figure 2.2: Microsoft Project Plan

IV. Risk Management

No	Risk content	Probability	Effect	Solution
#	**People Risk**			
1	Team member is not hard working, do not follow deadline.	Moderate	Significant	- Daily report - Pair review - Project manager should follow plan to remind.
2	Team member are sick, they cannot complete task under deadline.	Low	Significant	- Increase project team's working effort in "peace period". - Allow all team members clear about what others do, so that they can cover the tasks when necessary.
3	Conflictions between team members.	High	Not relevant	- Setup an open-talk environment in project team. - "Do not criticize" is set as a rule. - Organize teambuilding more often. - In some cases, manager must use his power to make decisions.
4	Poor experience makes plan late. Study new technologies have many difficult to apply for project	Moderate	Not relevant	- List tasks and check continuously. Evaluate quality and progress weekly. - Send email to other member to ask for help.
5	Bad communication breakdown can make changing time, work and delay plan	Moderate	Tolerable	- We need using words more clearly, talk with each other more, note and send email to confirm information.
#	**Technical Risk**			

6	We have not much knowledge in the framework and technique. Therefore, we have to study all of these things from the beginning. This work may take a lot of times or team may not resolve some technical problems.	Very high	Occurred	- Divided into technology research groups. - Exchange information and problem. - Send technical issues to supervisor who has experience to get support.
#	**Structure/ Process Risk**			
7	Underestimate project scope, tasks' difficulty level and risks' effectiveness.	Low	Serious	- Estimate project scope with supervisor and experience persons. - Assign task weight value to make task evaluation easier. Discuss in group about tasks' difficulty level. -Involve all team members in risk management process and reference to instructor's opinions.
#	**Requirement Risk**			
8	Not understanding the system's process , so we can make mistakes in describing the essential functions	Low	Potential	- Receive advice from experts - Develop prototypes and review prototypes with experts and supervisor
#	**Management Risk**			

9	Poor experience of management so that team makes plan unrealistically	Low	Serious	- Team leader will tightly co-operate with team members when planning. - Project team gets advice from supervisor about the planning and the plan need to be reviewed by supervisor.

V. Coding convention

Refers to coding convention of C#.Net. For more information please go to link:

http:// msdn.microsoft.com/codingconvention

VI. Quality Control

1. A quality demand and desired value

1.1 External quality characteristic

The quality demand managed in this project is as follows:

Quality characteristic	Quality demand
Functionality a. Finality b. Accuracy c. Standard conformity d. Security	a. All the functions defined in are realized. b. All the test cases of each test specification of a unit test, an integration test, and a system test are carried out. c. Ensure C# coding convention, Visual Studio. d. Follow security privacy in Apple document, all features access to user information are allowed by user.
Reliability a. Maturity nature b. Recoverability	a. All location information had been checked before providing to user to ensure high reliability. b. The system will work back within 2 days after occurring
Usability	- All design & features based on user experience, due to application is suitable with everyone uses iPhone and

	Android mobile.
Efficiency	The performance is high because optimizing code, focus to user demand.
Maintainability	Easy to maintain and extend in the future.
Schedule	Complete tasks on time.

1.2 Desired value.

The desired value in this project is set up as follows:

- High performance: application is not shock when loading game data.

2. Quality review plan

No.	Name	Note	Priority
Document quality review			
1	Report	Reports are satisfactory as supervisor asked.	High
iOS Application			
2	User Interface	UI in iOS application and Android application looks like Design in psd file.	High
	Features	All features must work well, high performance.	High

3. Test strategy

The test strategy of this project is shown below.

3.1 The kind of test

- The unit test for every program module (mainly enforcement of CR1).

- The integration test between program modules (mainly enforcement of CR1). - System test (mainly enforcement of CR1) ... Validation as the whole system.

3.2 Details of a test plan

- About each test will conduct after completing a feature. The developer will build application to device and tester will test and log bug if have.

- Each time detect bugs, develop has to fix by requirement.

- The bug will be closed after considering by team and supervisor.

3.3 Test environment

- 100% test on device (iOS 7 & Android 2.3 or higher).

Chapter 3 - Software Requirement Specification

I. User Requirement Specification

1. The purpose of systematization:

CR1 provide a mobile game application for consumer with purpose: user can play a game for relaxing and share glory moment with the other.

Furthermore, CR1 is needed to provide the service is developed and maintained, and then CR1 can provide some new things (update maps, cars...) and update function in expand project if project succeeded.

2. The mandatory functions of the application and important success factors:

CR1 will be developed to provide some main feature to user:

- ❖ View, change game settings
  - View maps (choose – unlock maps)
  - View cars (choose – unlock cars)
  - Choose music
  - Pause game
- ❖ Drive car
  - Control car (Turn – Use Boost Energy)
  - Pick up items
- ❖ View information
  - View Achievements
  - View Pick up coins
  - View Driven distance
  - View high scores
- ❖ Share facebook
- ❖ Purchase game coins
- ❖ Play with the other by Multiplayer

The priority of development:

- ❖ Play game single or multiplayer.
- ❖ Show details of game played
- ❖ Share achievements with the other over facebook share.

II. System Requirement Specification

1.External interface requirements

1.1 User interfaces

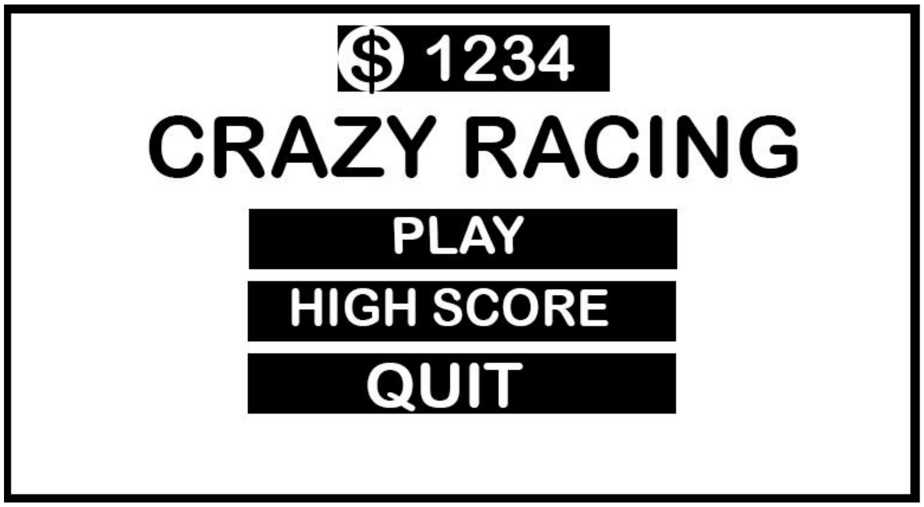


Figure 3.1: Main menu screen

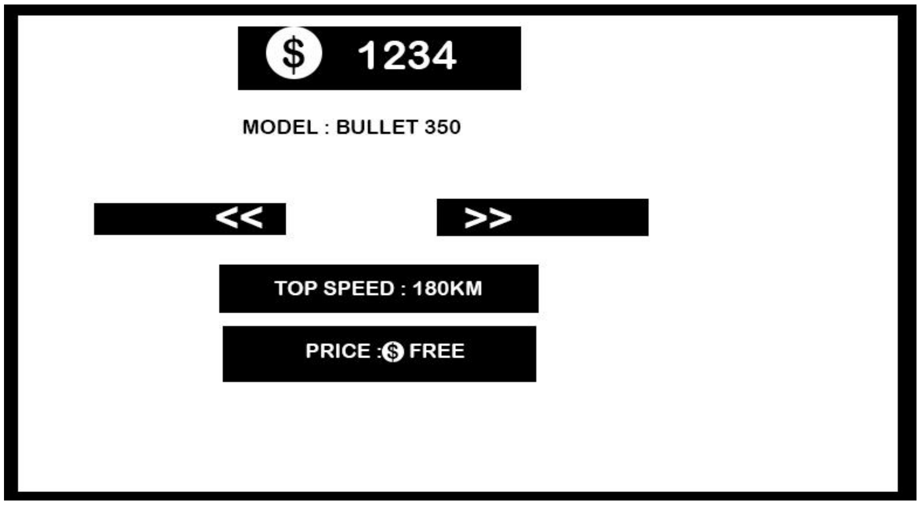


Figure 3.2: Main menu screen

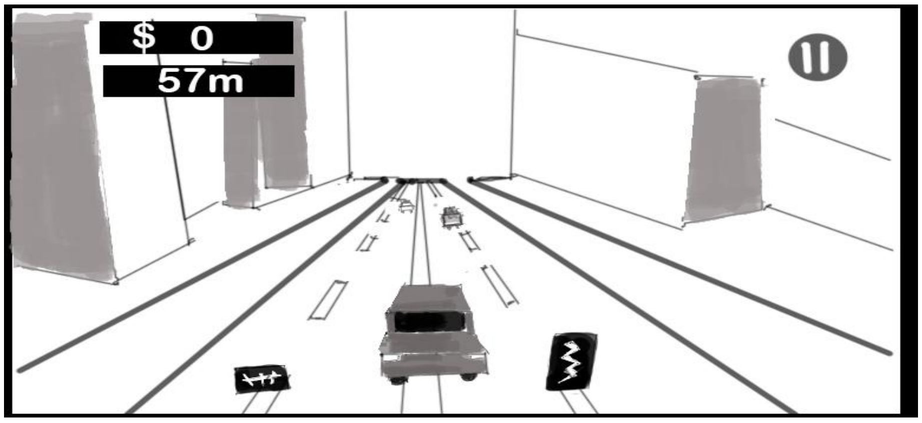
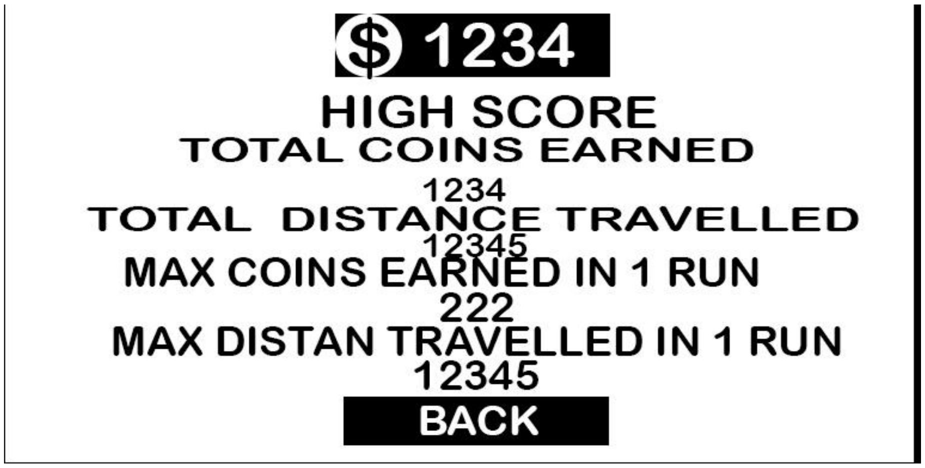


Figure 3.3: Game play screen



$ 1234
HIGH SCORE
TOTAL COINS EARNED
1234
TOTAL DISTANCE TRAVELLED
12345
MAX COINS EARNED IN 1 RUN
222
MAX DISTAN TRAVELLED IN 1 RUN
12345
BACK

Figure 3.4: Highscore screen

1.2 Hardware interfaces

- Android device 2.3/iOS device 6.1 or higher.
- Both of types must support wifi connection.

2. Main flow overviews

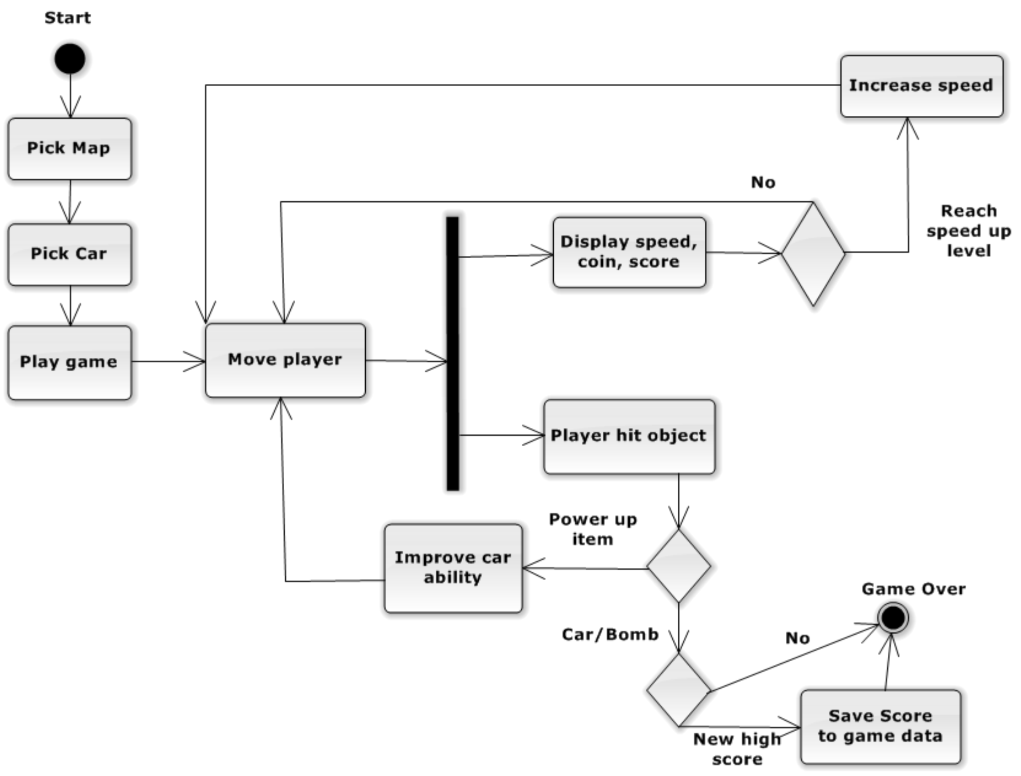


Figure 3.5: Mainflow

3. System Features

3.1 Overall use case diagram

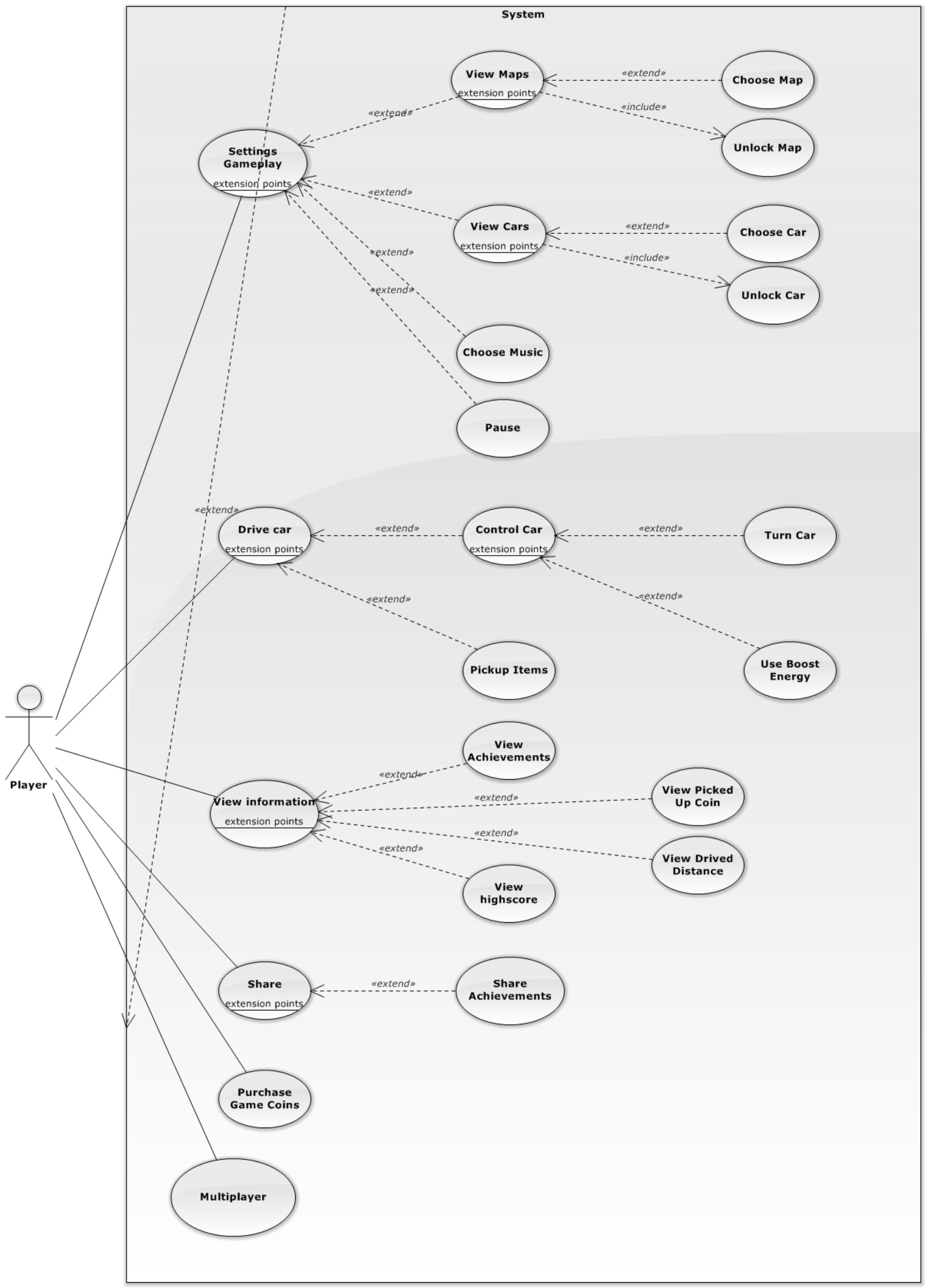


Figure 3.6: Usecase diagram

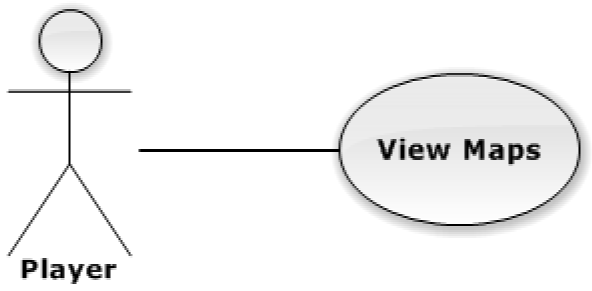
3.2 Function definition

The game focus on the iOS and Android operating system that will interact directly with user by operation of user.

No	Function name	Description
1	View Maps	Display all maps to user.
2	Choose Map	User can choose a map for racing.
3	Unlock Map	User can unlock a new map by coins.
4	View Cars	Display all cars to user.
5	Choose Car	User can choose a car for racing. In this version of game, only support to change skin of car, does not include engine, tire etc.
6	Unlock Car	User can unlock a new car by coins.
7	Choose music	User can choose name of song, which is played in game.
8	Pause	Pause the game.
9	Turn Car	Control car to turn right, left.
10	Use Boost Energy	Control car to move with the highest speed.
11	Pickup Items	Control car to pick up items on the street. These items may be: - Invisible boost (can run through another car). - Double coins. - Magnet, which automatic gets coins from the street. - Bazooka: hit helicopter by missile. - Supper: hit all traffic cars with affects. - Instant nitrous: full energy for boost speed.
12	Share Achievements	User can share achievements in game: distance, coins.
13	Share Pictures	User can share pictures in game.
14	View Achievements	After playing game, user can see all of information of this game, which includes: - Achievements.
15	View Picked Up Coin	
16	View Driven Distance	

		- Coins. - Distance.
17	View High Score	In main menu, user can see the highest score of himself/herself.
18	Purchase Game Coins	User can buy one of three packages of coins by purchase via apple/ google store.
19	Connect to another Player Multiplayer	User can play in multiplayer mode with another.

3.2.1 View Maps



VIEW MAPS – SPECIFICATION			
Use case No.	UC001	**Use case Version**	0.1
Use case Name	View Maps		
Author	Trần Quốc Duy		
Date	30/01/2015	**Priority**	Normal

Actor:
Player.
Summary:
This use case allows user to see all maps that include available maps and unlock maps.
Goal:
Display all maps correctly.
Triggers:
After user have chosen music, the “View Maps Screen” displays.
Preconditions:
User opened application successfully, chosen car and music.
Post-Conditions:
On success: Display all maps that include available maps and unlock maps.
On failure: N/A.

Main Success Scenario:

Step	User Action	System Response
1.	Choose a song.	
2.		Navigate to “View maps” screen that displays all the map that include available and unavailable maps.

Alternative Scenario: N/A.

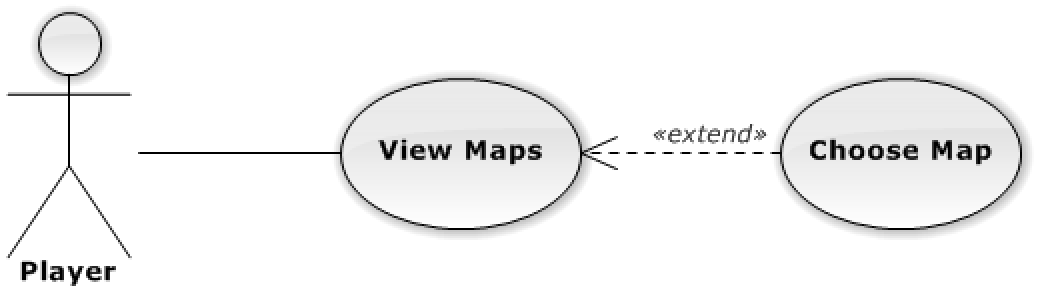
Exceptions: N/A.

Relationships: N/A.

Business Rules:

- Available maps have the bold color, opposite the unlock maps have light color with the lock icon on the top.

3.2.2 Choose Map



CHOOSE MAP – SPECIFICATION			
Use case No.	UC002	**Use case Version**	0.1
Use case Name	Choose Map		
Author	Trần Quốc Duy		
Date	30/01/2015	**Priority**	Normal

Actor:
Player.
Summary:
This use case allows user to select a map (only available car) to use in the race.
Goal:
The chosen map must be display correctly in the race.
Triggers:
User select the wanted map image.
Preconditions:

All maps display correctly.

Post-Conditions:

On success: Navigate user to choose music screen.

On failure: N/A.

Main Success Scenario:

Step	**User Action**	**System Response**
1.	Touch the map image.	
2.		Navigate user to “Starting race” screen.

Alternative Scenario: N/A.

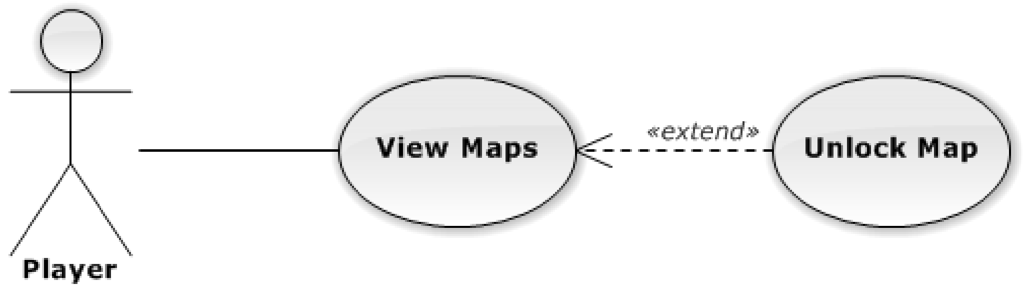
Exceptions: N/A.

Relationships: View Maps use case.

Business Rules:

- User can only select available maps.

3.2.3 Unlock Map



UNLOCK MAP – SPECIFICATION			
Use case No.	UC003	**Use case Version**	0.1
Use case Name	Unlock Map		
Author	Trần Quốc Duy		
Date	30/01/2015	**Priority**	Low

Actor:

Player.

Summary:

This use case allows user to unlock a map by coins (make a map available to select) to use in the race.

Goal:

The unlocked map become available map.

Triggers:
User select the unlock icon of locked map to unlock it.
Preconditions:
All maps display correctly.
Post-Conditions:
On success: The unlocked map become available map with bold color.
On failure: N/A.
Main Success Scenario:

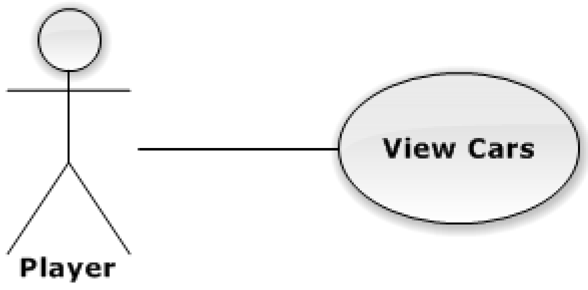
Step	User Action	System Response
1.	Touch the unlock icon of unavailable map image.	
2.		Display message that ask user pay corresponding coin for unlocking this map.
3.	User select OK	
4.		The map is unlocked.

Alternative Scenario:

Step	User Action	System Response
1.	Touch the unlock icon of unavailable map image but the coin is not enough.	
2.		Show message that notify user is not enough coins.
3.	User select OK	
4.		Message hide. The chosen map still locked.

Exceptions: N/A.
Relationships: View Maps use case.
Business Rules:
- User can only unlock the locked map.
- To unlock, user's coins must be equal or higher than necessary coins

3.2.4 View Cars



VIEW CARS – SPECIFICATION			
Use case No.	UC004	**Use case Version**	0.1
Use case Name	View Cars		
Author	Trần Quốc Duy		
Date	30/01/2015	**Priority**	Normal

Actor:
Player.
Summary:
This use case allows user to see all cars that include available cars and unlock cars.
Goal:
Display all cars correctly.
Triggers:
After user choose “Start” button, the “View Car Screen” displays.
Preconditions:
User opened application successfully.
Post-Conditions:
On success: Display all cars that include available cars and unlock cars.
On failure: N/A.
Main Success Scenario:

Step	User Action	System Response
1.	Press “Start” button.	
2.		Display all cars

Alternative Scenario: N/A.

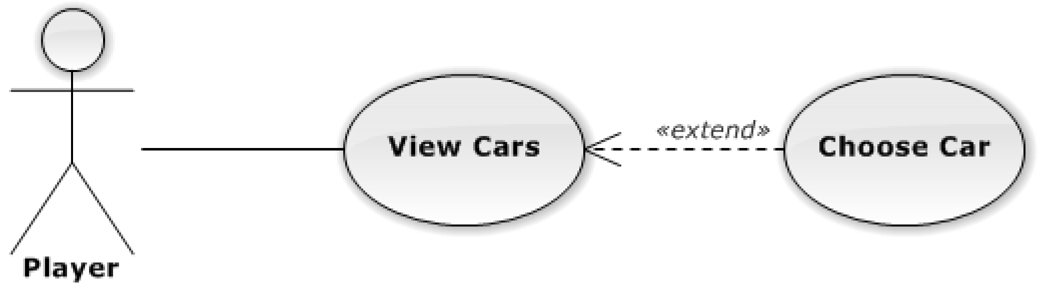
Exceptions: N/A.

Relationships: N/A.

Business Rules:
- Available cars have the bold color, opposite the unlock car have light color with the lock

icon on the top.

3.2.5 Choose Car



CHOOSE CAR – SPECIFICATION			
Use case No.	UC005	**Use case Version**	0.1
Use case Name	Choose Car		
Author	Trần Quốc Duy		
Date	30/01/2015	**Priority**	Normal

Actor:
Player.
Summary:
This use case allows user to select a car (only available car) to play in the race.
Goal:
The chosen car must be display correctly in the race.
Triggers:
User select the wanted car image.
Preconditions:
All cars display correctly.
Post-Conditions:
On success: Navigate user to choose music screen.
On failure: N/A.
Main Success Scenario:

Step	User Action	System Response
1.	Touch the car image.	
2.		Navigate user to choose music screen

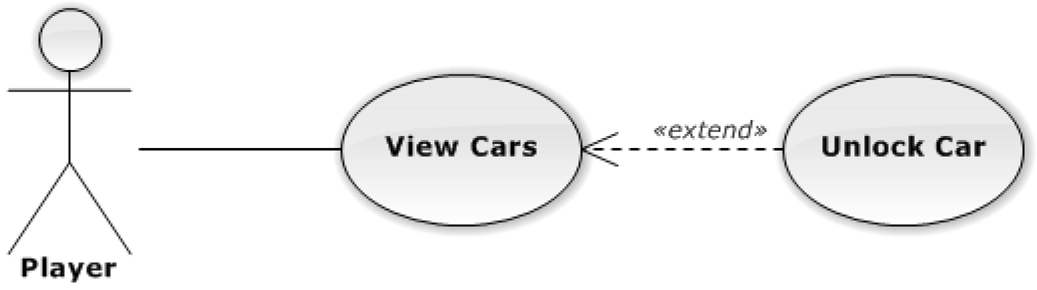
Alternative Scenario: N/A.

Exceptions: N/A.

Relationships: View Cars use case.
Business Rules:

- User can only select available car.

3.2.6 Unlock Car



UNLOCK CAR – SPECIFICATION			
Use case No.	UC006	**Use case Version**	0.1
Use case Name	Unlock Car		
Author	Trần Quốc Duy		
Date	30/01/2015	**Priority**	Low

Actor:
Player.
Summary:
This use case allows user to unlock a car by coins (make a car available to select) to play in the race.
Goal:
The unlocked car become available car.
Triggers:
User select the unlock icon of locked car to unlock it.
Preconditions:
All cars display correctly.
Post-Conditions:
On success: The unlocked car become available car with bold color.
On failure: N/A.
Main Success Scenario:

Step	User Action	System Response
1.	Touch the unlock icon of unavailable car image.	
2.		Display message that ask user pay corresponding coin for unlocking this car.

3.	User select OK	
4.		The car is unlocked.

Alternative Scenario:

Step	User Action	System Response
1.	Touch the unlock icon of unavailable car image but the coin is not enough.	
2.		Show message that notify user is not enough coins.
3.	User select OK	
4.		Message hide. The chosen car still locked.

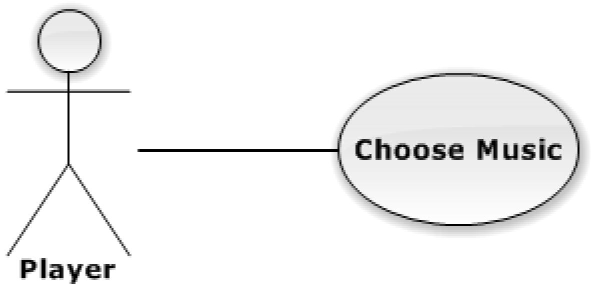
Exceptions: N/A.

Relationships: View Cars use case.

Business Rules:

- User can only unlock the locked car.
- To unlock, user's coins must be equal or higher than necessary coins

3.2.7 Choose Music



CHOOSE MUSIC – SPECIFICATION			
Use case No.	UC007	**Use case Version**	0.1
Use case Name	Choose Music		
Author	Trần Quốc Duy		

Date	30/01/2015	Priority	Low

Actor:
Player.
Summary:
This use case allows user to select a song that playing in the race.
Goal:
The chosen music must be play while user playing game.
Triggers:
User select a song in the list.
Preconditions:
List of music display correctly
Post-Conditions:
On success: Navigate user to choose music screen.
On failure: N/A.
Main Success Scenario:

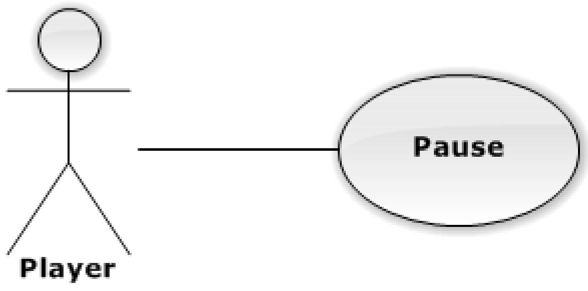
Step	User Action	System Response
1.	Select a cars.	
2.		Navigate user to “Music” screen.

Alternative Scenario: N/A.

Exceptions: N/A.

Relationships:
Business Rules: N/A

3.2.8 Pause



PAUSE – SPECIFICATION			
Use case No.	UC008	**Use case Version**	0.1
Use case Name	Pause		

Author	Trần Quốc Duy		
Date	30/01/2015	**Priority**	Low

Actor: Player.
Summary: This use case allows user to pause the game while playing.
Goal: Keep status of game, and resume correctly when user comeback the game

Triggers:
User select "Back" or "Menu" button of device.
Preconditions:
User's is playing game
Post-Conditions:
On success: All the status of object in game is keeping, storing.
On failure: N/A.
Main Success Scenario:

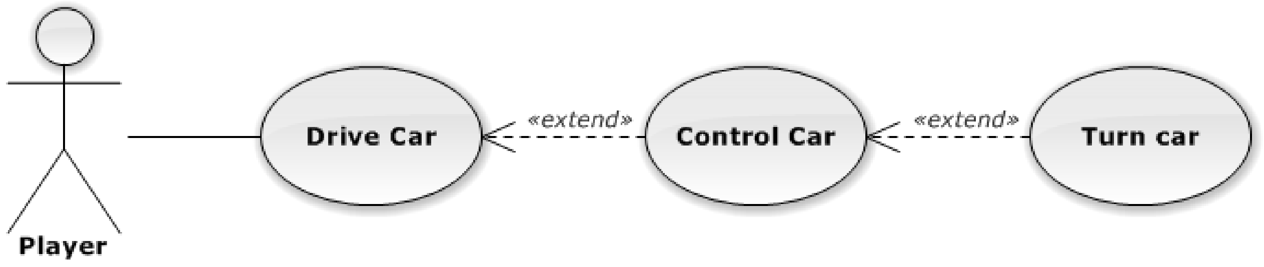
Step	User Action	System Response
1.	User select "Back" or "Menu" button of device.	
2.		Pause the game, keep all status of the all game object.

Alternative Scenario: N/A.

Exceptions: N/A.

Relationships:
Business Rules: N/A

3.2.9 Turn Car



TURN CAR – SPECIFICATION			
Use case No.	UC009	**Use case Version**	0.1

Use case Name	Turn Car		
Author	Nguyễn Cao Cường		
Date	29/01/2015	**Priority**	High

Actor:

Player.

Summary:

This function allow player to turn right or left.

Goal:

Turn car to evade the obstacles or get power-up items.

Triggers:

Running screen.

Preconditions:

After choose options and go to game play screen, in running.

Post-Conditions:

On success: Car turn right or left

On failure: N/A.

Main Success Scenario:

Step	User Action	System Response
1.	Skew the mobile to right or left	
2.		Car turn right or left

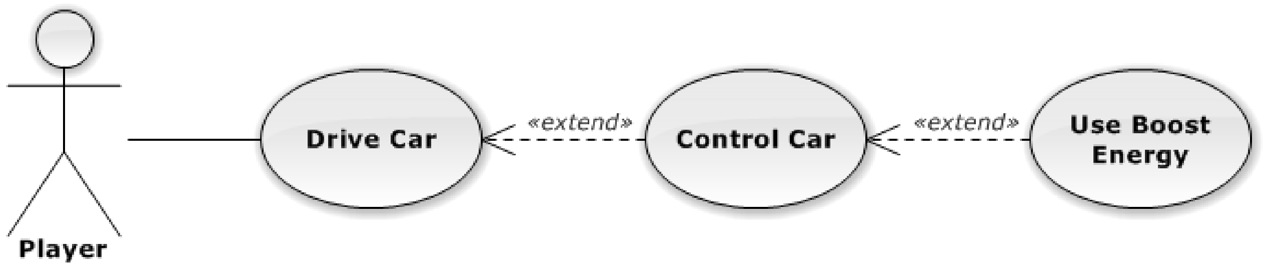
Alternative Scenario: N/A.

Exceptions: N/A.

Relationships: N/A.

Business Rules: N/A

3.2.10 Use Boost Energy



USE BOOST ENERGY – SPECIFICATION			
Use case No.	UC010	**Use case Version**	0.1

Use case Name	Use Boost Energy		
Author	Nguyễn Cao Cường		
Date	29/01/2015	**Priority**	Normal

Actor:
Player

Summary:
This function allow player to accelerate by using boost energy. There's nitrogen pool on the screen, player will fill up this pool by earning coin on the street. When player use boost energy, nitrogen gradually reduced over time using.

Goal:
Accelerate to get high speed.

Triggers:
Running screen.

Preconditions:
After choose options and go to game play screen, in running.

Post-Conditions:
On success: Accelerate
On failure: Do nothing. See in Exception

Main Success Scenario:

Step	User Action	System Response
1.	Player tap on "Boost Energy" button	
2.		Car get high speed

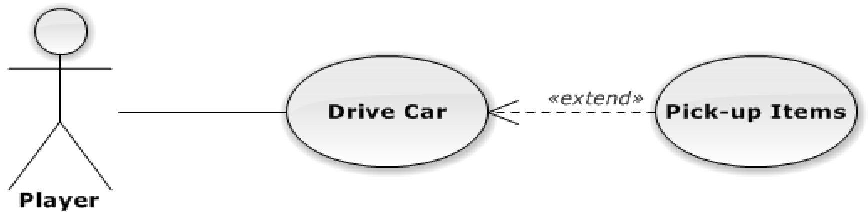
Alternative Scenario: N/A.

Exceptions:

Step	User Action	System Response
1.	No nitrogen in pool	No accelerate

Relationships: N/A.
Business Rules: N/A

3.2.11 Pickup Items



PICK UP ITEMS – SPECIFICATION			
Use case No.	UC011	**Use case Version**	0.1
Use case Name	Pick-up Items		
Author	Nguyễn Cao Cường		
Date	29/01/2015	**Priority**	Normal

Actor:
Player
Summary:
This function allow player to pick power-up items.
Goal:
Pick items up and get effect.
Triggers:
Running screen.
Preconditions:
After choose options and go to game play screen, in running.
Post-Conditions:
On success: Get effect
On failure: N/A.
Main Success Scenario:

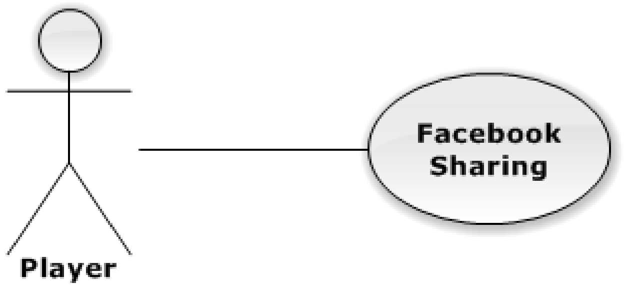
Step	User Action	System Response
1.	Player skew the screen to pick up items	
2.		Car will have the effect corresponding to item that picked up

Alternative Scenario: N/A.

Exceptions: N/A.

Relationships: N/A.
Business Rules: N/A

3.2.12 Share Achievements



FACEBOOK SHARING – SPECIFICATION			
Use case No.	UC012	**Use case Version**	0.1
Use case Name	Facebook sharing		
Author	Nguyễn Cao Cường		
Date	29/01/2015	**Priority**	Normal

Actor:

Player

Summary:

This function allow player to share achievement.

Goal:

Share high achievement on Facebook

Triggers:

Score Board after game over

Preconditions:

Play game, running until game-over

Post-Conditions:

On success: Share achievement on Facebook

On failure: Do nothing, show message. See in Exception.

Main Success Scenario:

Step	User Action	System Response
1.	Player tap on "Facebook Share" button on Scored Board screen	
2.		Call API of Facebook, post achievement on the FB wall if player logged in.

Alternative Scenario: N/A.

Exceptions:

Step	User Action	System Response

1.	Network problem condition	Display message: “No network connected”.

Relationships: N/A.
Business Rules: N/A

3.2.13 View Driven Distance

3.2.14 View High Score

VIEW DRIVEN DISTANCE – SPECIFICATION			
Use case No.	UC013	**Use case Version**	0.1
Use case Name	View Driven Distance		
Author	Tôn Thất Hoàng Triều		
Date	28/01/2015	**Priority**	Normal

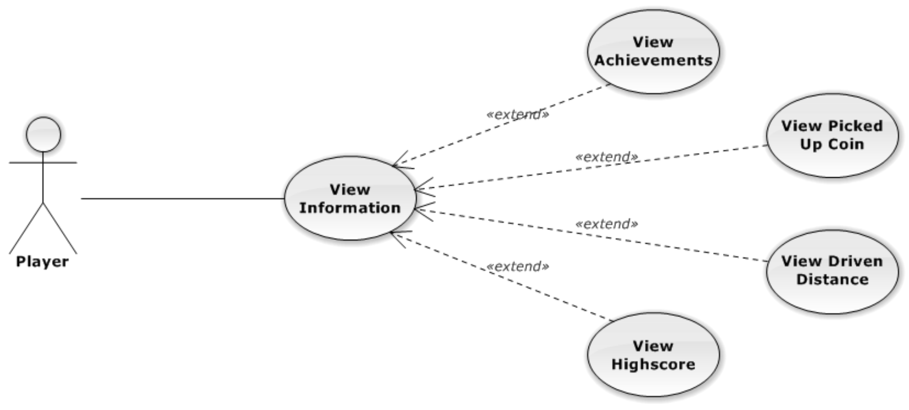
Actor:
Player.
Summary:
This use case allows user to view the distance his/her have been driven.
Goal:
Display the driven distance.
Triggers:
Overgame – the game is stopped (hit other car).
Preconditions:
User play game and the game is over.
Post-Conditions:
On success: Display the distance, that means travelled distance, unit is m(meters).
On failure: N/A.
Main Success Scenario:

Step	**User Action**	**System Response**
1.	Let the game over.	
		Display the driven distance.

Alternative Scenario: N/A.

Exceptions: N/A.

Relationships: N/A.
Business Rules: N/A.



VIEW HIGHSCORE – SPECIFICATION			
Use case No.	UC014	**Use case Version**	0.1
Use case Name	View highscore		
Author	Tôn Thất Hoàng Triều		
Date	29/01/2015	**Priority**	Normal

Actor:

Player

Summary:

This use case allows user to view the highest score of himself/herself.

Goal:

Display the highest score, which includes earned coin, driven distance.

Triggers:

User presses "High score" button on main menu.

Preconditions:

User opened app, pressed "High score" button on main menu.

Post-Conditions:

On success: Display the highest score.

On failure: N/A.

Main Success Scenario:

Step	User Action	System Response
1.	User press "High score" button.	
2.		Display the best record of user.

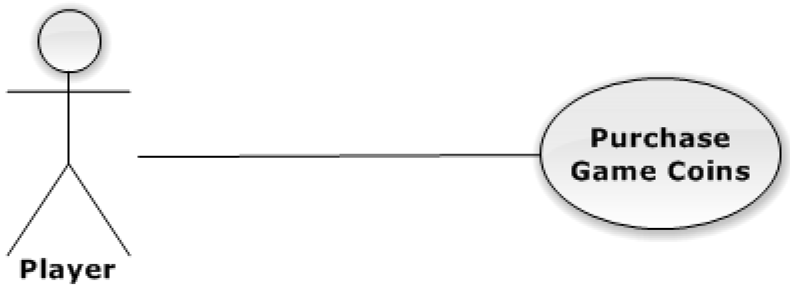
Alternative Scenario: N/A.

Exceptions: N/A.

Relationships: N/A.

Business Rules: N/A.

3.2.15 Purchase Game Coins



PURCHASE GAME COINS – SPECIFICATION			
Use case No.	UC015	**Use case Version**	0.1
Use case Name	Purchase Game Coins		
Author	Tôn Thất Hoàng Triều		
Date	29/01/2015	**Priority**	Normal

Actor:

Player.

Summary:

This use case allows user to buy coin via apple store or google store account.
We have 3 packages of coin to let user purchase:
- 2,000 coins: 0.99$.
- 10,000 coins: 2.99$.
- 50,000 coins: 9.99$.

Goal:

Let user change money in account to coins in game.

Triggers:

User chooses one of 3 packages.

Preconditions:

User must login to apple/ google account to purchase.

Post-Conditions:

On success: User change money to coins successfully.
On failure: N/A.

Main Success Scenario:

Step	User Action	System Response
1.	Choose one package of coins.	
		Transfer user to apple/google store page via browser.
2.	Confirm a changing of money to coins with apple/google store.	

		Receive response from store, give user coins base on her/his chosen package of coins.

Alternative Scenario: N/A.

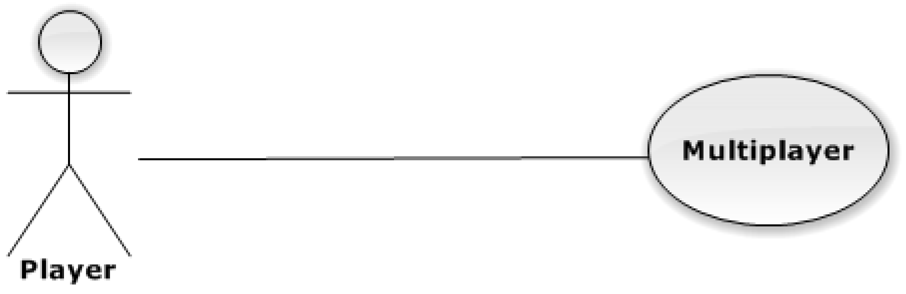
Step	User Action	System Response
1.	Choose one package of coins.	
		Transfer user to apple/google store page via browser.
2.	Does not confirm a changing of money to coins with apple/google store.	
		Receive response from store, give user nothing.

Exceptions: N/A.

Relationships: N/A.

Business Rules: N/A.

3.2.16 Multiplayer



MULTIPLAYER – SPECIFICATION			
Use case No.	UC016	**Use case Version**	0.1
Use case Name	Multiplayer		
Author	Tôn Thất Hoàng Triều		
Date	29/01/2015	**Priority**	Normal

Actor:

Player.

Summary:

This use case allows user to connect to another player via wifi.
Then they will play in multiplayer mode.

Triggers:

Choose multiplayer mode option in main menu.

Preconditions:

User opened app, went to main menu.

Post-Conditions:

On success: User connects successfully with the other.
On failure: N/A.

Main Success Scenario:

Step	User Action	System Response
1.	Choose multiplayer mode in main menu.	
		Generate a code to send to another player.
2.	2nd user must input a generated code from 1st user.	
		Let users play game in multiplayer mode.

Alternative Scenario: N/A.

Step	User Action	System Response
1.	Choose multiplayer mode in main menu.	
		Generate a code to send to another player.
2.	2nd user inputs a generated code different from 1st user code.	
		Show popup to notice to user wrong code. See [Exceptions 1]

Step	User Action	System Response

1.	Choose multiplayer mode in main menu.	
		Generate a code to send to another player.
2.	2nd user inputs a generated code from 1st user code.	
		Let users play game in multiplayer mode. See [Exceptions 2]

Exceptions:

No	User Action	System Response
1	Enter wrong code.	Display message: "Wrong code. Please check carefully".
2.	Play game in network problem conditions.	Display message: "Sorry, no network connecting. Please check your network".

Relationships: N/A.
Business Rules: N/A.

3.3 Software system attributes

Performance

Requirements relating to Performance	
No.	**Requirement**
1.	Android version 2.3 or higher, chipset 0.4GHz or higher, RAM 256 MB or higher, phone memory/SDD card, wifi connection. Response time for CR system should meet following: - For the first time to open app, it take no longer than 5 seconds. - For all validation data logic, the response time shouldn't take than 2 seconds
2	With mentioned server above, CR system should work smoothly in low RAM memory condition.

Scalability

Requirements relating to Scalability	
No.	**Requirement**
1.	It must scale to the expected to run in 2 mobile device at same time to play multiplayer mode.

Security

Requirements relating to Security	
No.	**Requirement**
1.	Just use standard authentication and authorization mechanism of apple store and google store.

Portability

Requirements relating to Portability	
No.	**Requirement**
1.	For the up-coming release, the CR is expected to work with Android OS version 2.3 or higher, iOS version 6.1 or higher.

Error Handling

Requirements relating to Error Handling	
No.	**Requirement**
1.	Proactive notification of problems. System must provide sufficient context in the notification to assist in the diagnosis and repair of the problem. Varying levels of notification will be needed for different classes of error instances: logging errors to log files, logging errors to event viewer, sending messages.

Infrastructure

Requirements relating to Infrastructure	
No.	**Requirement**
1.	All services inside CR are expected to work with local database.

Support& Supportability

Requirements relating to Support	
No.	**Requirement**
1.	Still maintain and update new version into store.

Reliability

Requirements relating to Reliability	
No.	**Requirement**
1.	Availability: The CR server is expected to run all the time 24 hours a day and 7 day a week without crash.

Design Constraints

Requirements relating to Design	
No.	**Requirement**
1.	The design must take this requirement into consideration for everything that the system may do and how this could be supported

4.Game Description

4.1 Game objects

4.1.1 Cars

- Player’s car:

+ Default

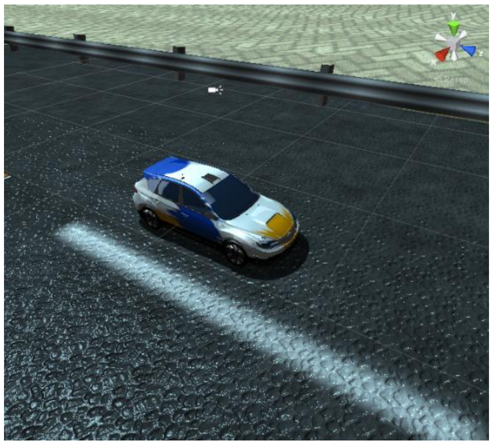


Figure 3.7: Default car

+ Blue car:

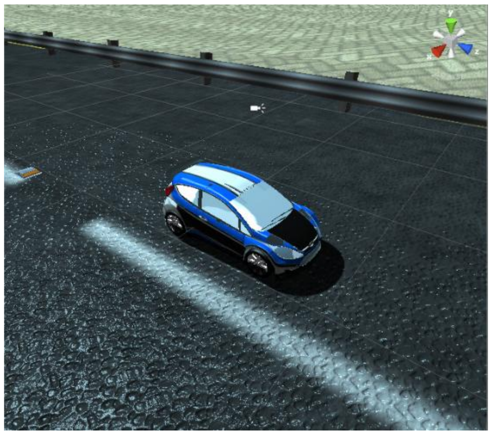


Figure 3.8: Blue car

+ Green car:

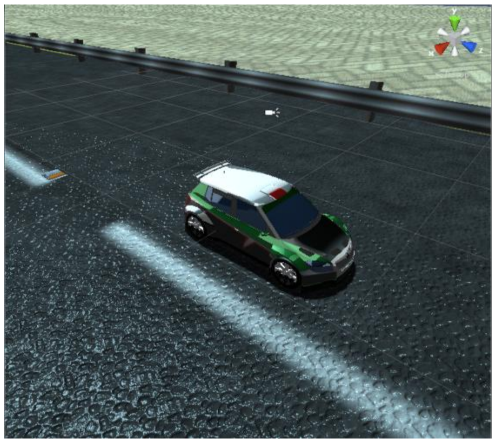


Figure 3.9: Green car

+ Red car:

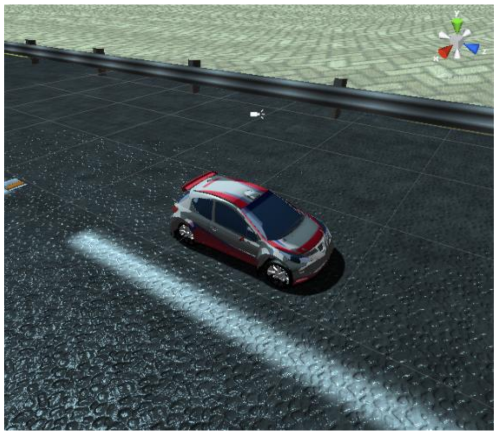


Figure 3.10: Red car

	Car type		
	Blue	Green	Red
Turn speed	22	22	22
Car speed	6	7	8
Max percent speed	300	360	420
Percent speed	100	100	100
Distance level up	500	400	300
Titl	0.3	0.3	0.3

Wheel speed	800	800	800
Magnet power time	7	7	8
Double coin power time	7	7	8
Ghost power time	7	7	8

- Traffic cars:

+ Ambulance:

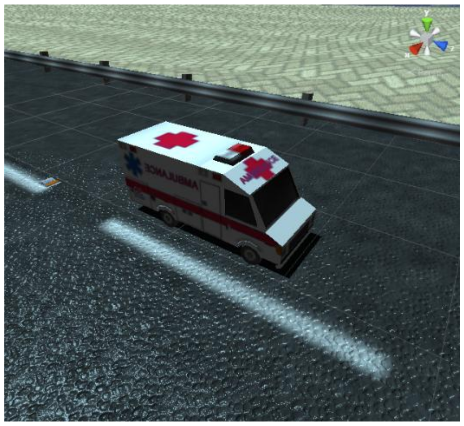


Figure 3.11: Ambulance

+ Bus:

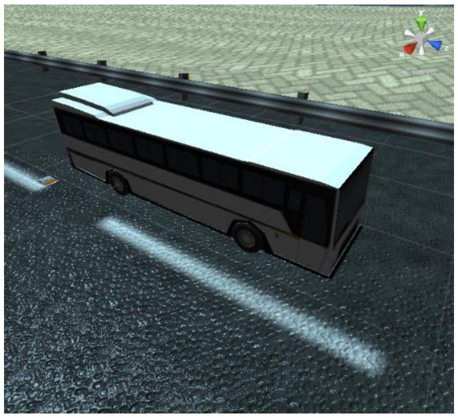


Figure 3.12: Bus

+ Empty truck:

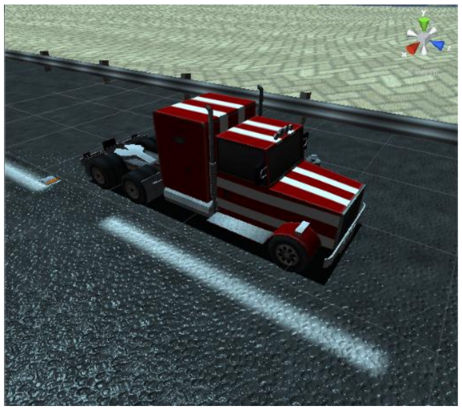


Figure 3.13: Empty truck

+ Jeep:

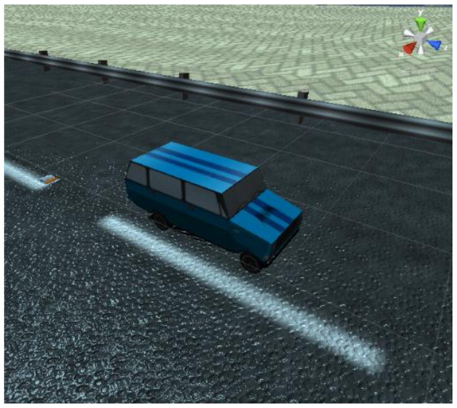


Figure 3.14: Jeep

+ Mini truck:

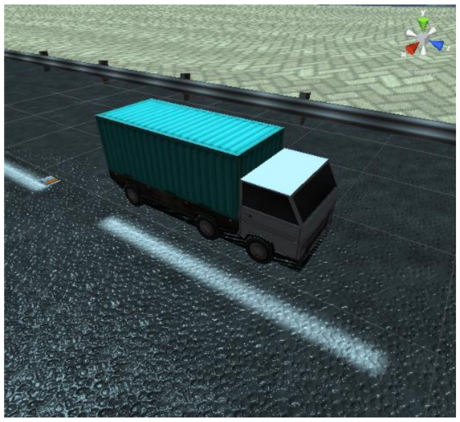


Figure 3.15: Mini truck

+ Oil tanker:

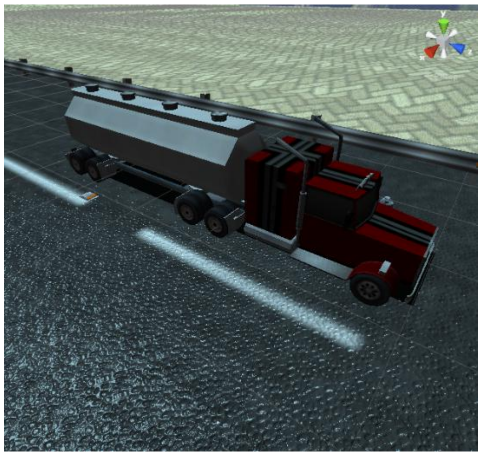


Figure 3.16: Oil tanker

+ Small truck:

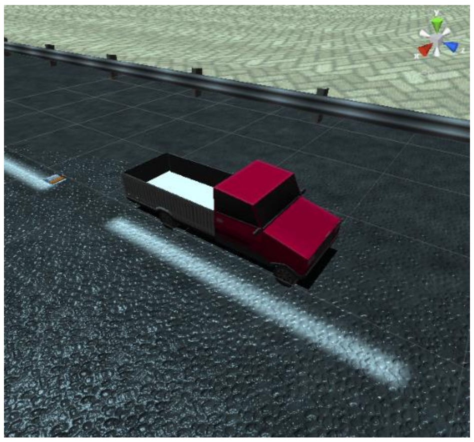


Figure 3.17: Small truck

+ Xmas car:

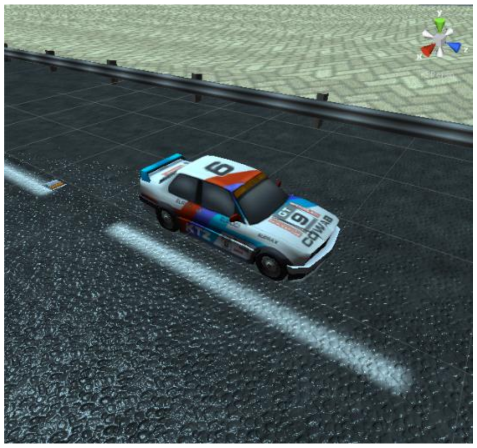


Figure 3.18: Xmas car

4.1.2 Items

- Coin: player can get coins on street, which are used to boosting car's speed up.

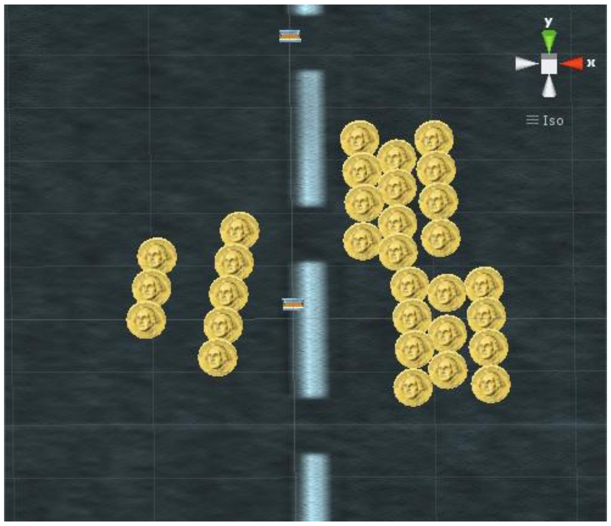


Figure 3.19: Coins

- Bazooka: pick it on street and missile will hit helicopter automatically.

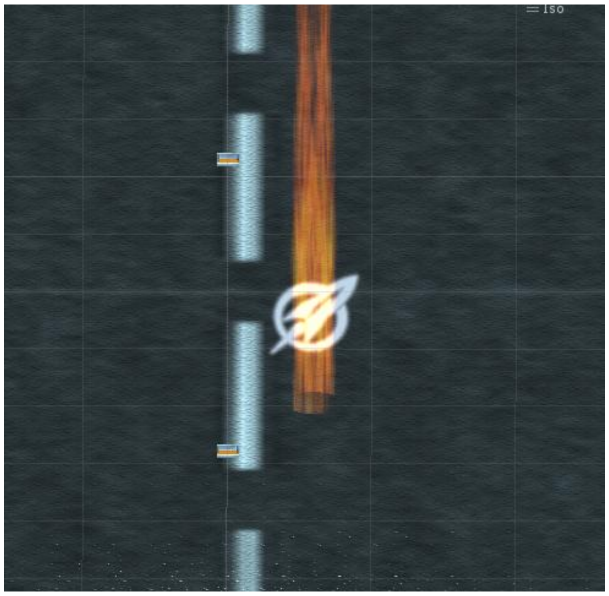


Figure 3.20: Bazooka

- Ghost: if player pick it on street, player’s car can go through traffic cars in a moment.

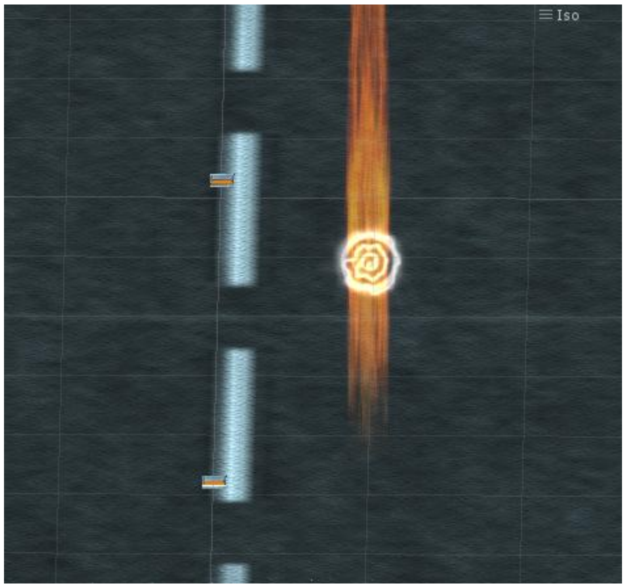


Figure 3.21: Ghost

- Magnet: player’s car can automatic pick coin on street, without approaching coins.

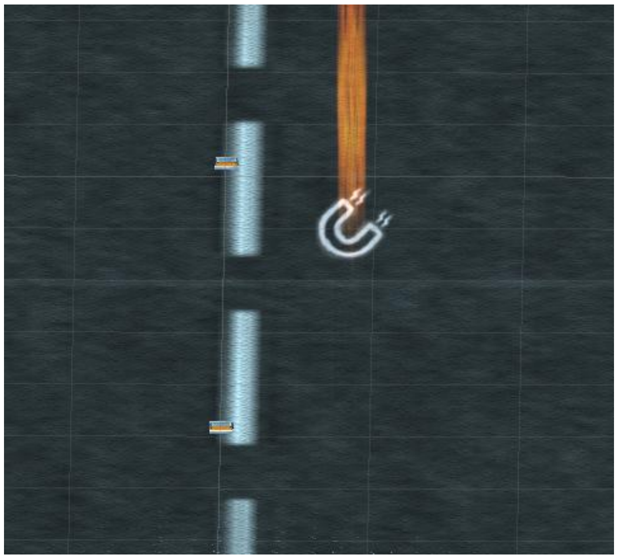


Figure 3.22: Magnet

- Double coin: double the number of picked up coins.

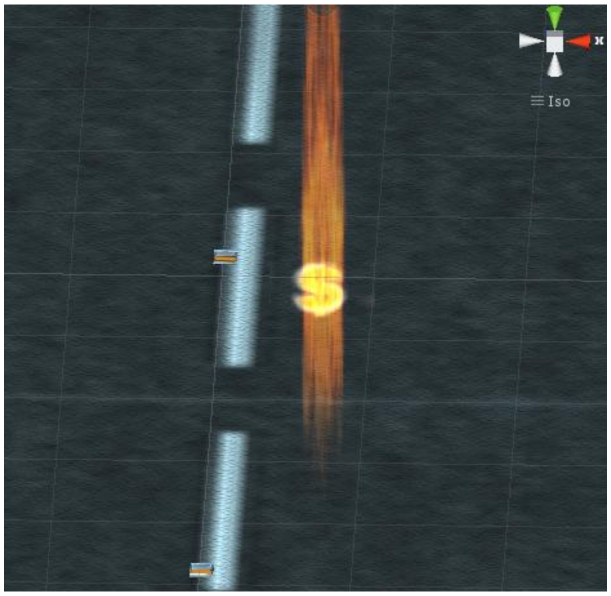


Figure 3.23: Double coin

- Instant nitrous: energy for boost car’s speed can full immediately if user pick it up.

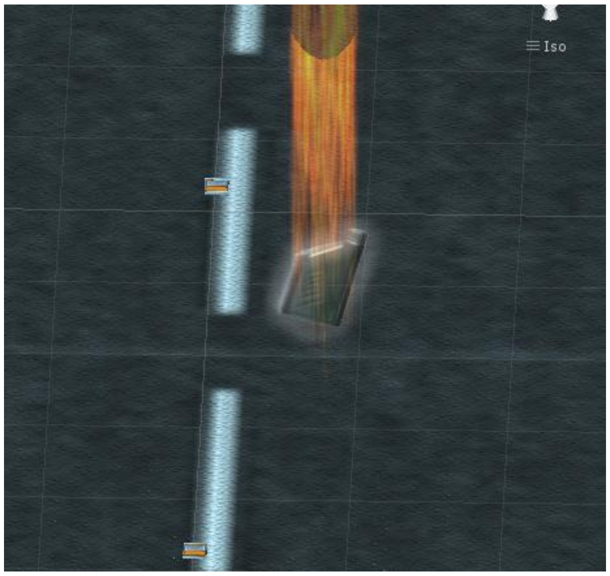


Figure 3.24: Instant nitrous

4.1.3 Maps

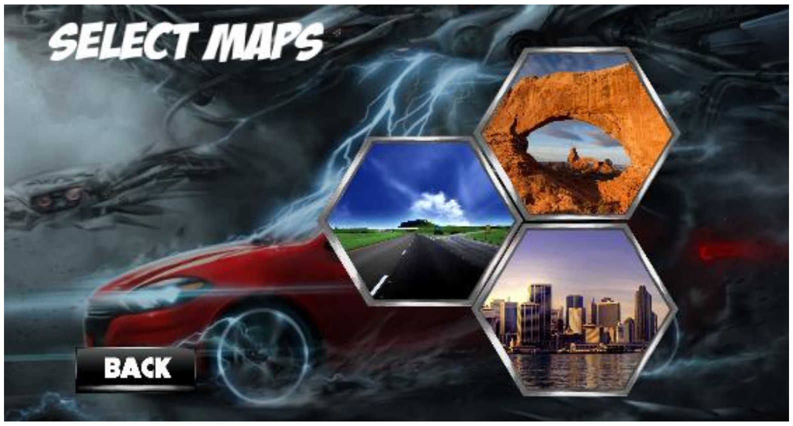


Figure 3.25: Maps

- Map 1: City

- Map 2: Country

- Map 3: Desert

4.1.4 Helicopter

- Helicopter will attack player car by missle. This process provides player car attack point on the street, user have to control the car without run on attack point, if user hits this point, the car will be destroyed.

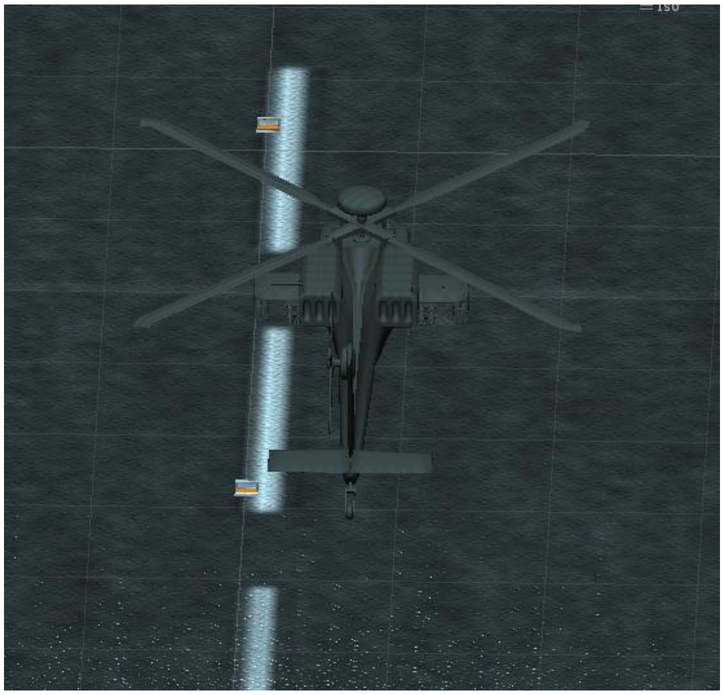


Figure 3.26: Helicopter

- Attack point:

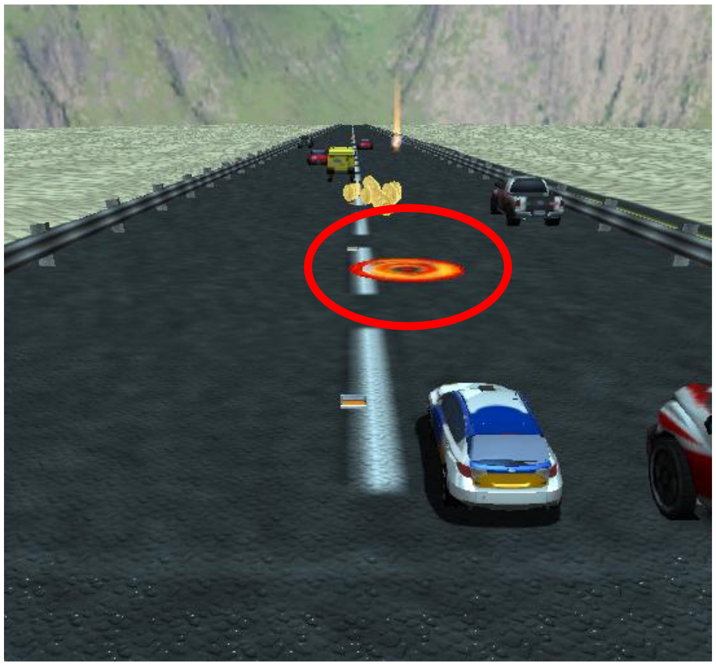


Figure 3.27: Attack point

Chapter 4 - Software Design Description

I. Introduction

This document describes the architecture design, the detail design and the class design. The architecture design contains the overall architecture design of the system and its subs – system. The detail design represents the structure and the behaviors of the component. The class design describes detail design of class and relationship of them. Each of the following sections are summarized below:

. Software architecture design

. Details description of components

II. Software architecture Design

1. Unity engine

First, we should follow some components of Unity3d structure

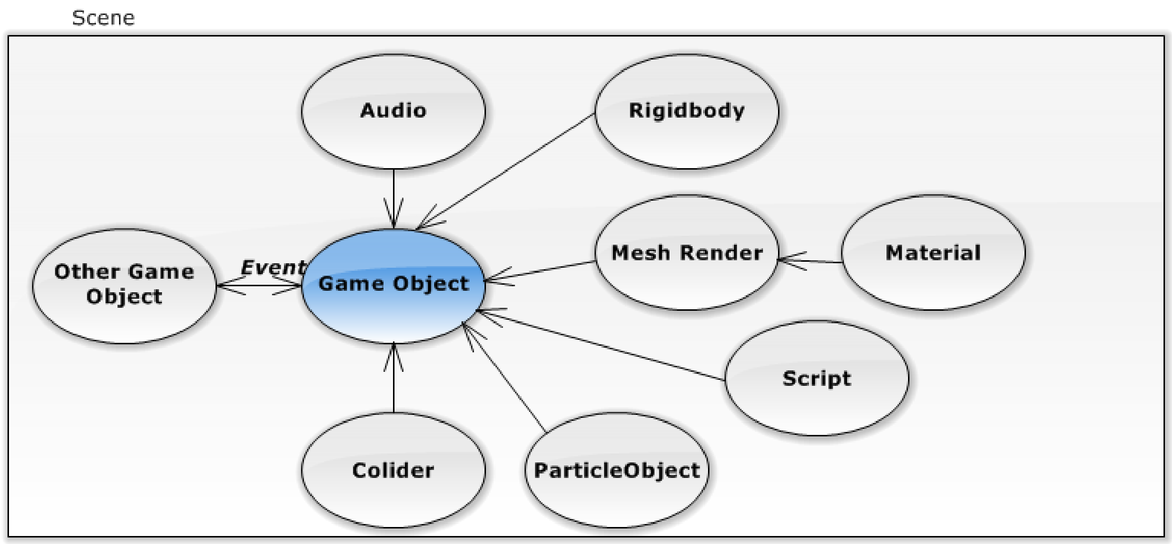


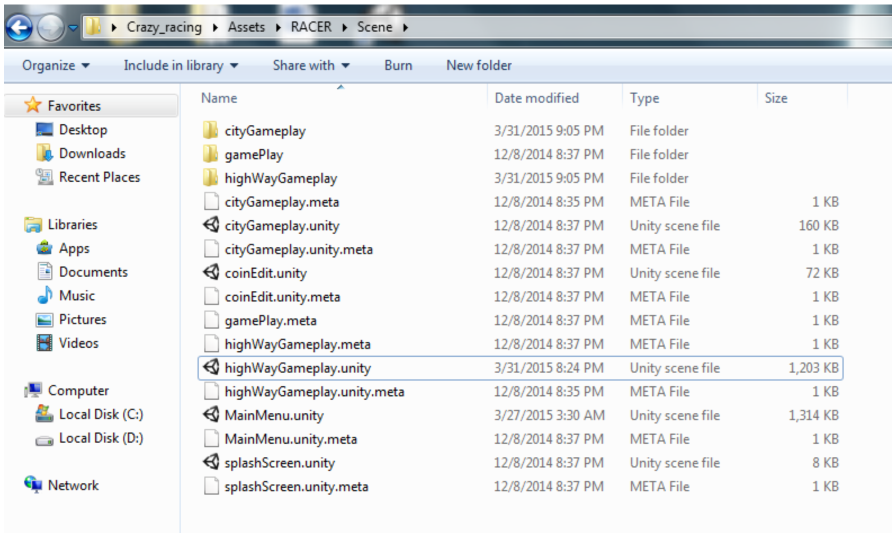
Figure 4.1: Game object's component

1.1 Scene

In Unity, you should think of scenes as individual levels, or areas of game content—though some developers create entire games in a single scene, such as, puzzle games, by dynamically loading content through code. By constructing your game with many scenes, you'll be able to distribute loading times and test different parts of your game individually. New scenes are often used separately to a game scene you may be working on, in order to prototype or test a piece of potential gameplay.

A scene is created by one more many game objects. A game object is created by components such as: collider, rigridbody, audio, scripts, particle object, mesh render and script. Game objects communicate by event.

Any currently open scene is what you are working on, as no two scenes can be worked on simultaneously. Scenes can be manipulated and constructed by using the Hierarchy and Scene views.



Name	Date modified	Type	Size
cityGameplay	3/31/2015 9:05 PM	File folder	
gamePlay	12/8/2014 8:37 PM	File folder	
highWayGameplay	3/31/2015 9:05 PM	File folder	
cityGameplay.meta	12/8/2014 8:35 PM	META File	1 KB
cityGameplay.unity	12/8/2014 8:37 PM	Unity scene file	160 KB
cityGameplay.unity.meta	12/8/2014 8:37 PM	META File	1 KB
coinEdit.unity	12/8/2014 8:37 PM	Unity scene file	72 KB
coinEdit.unity.meta	12/8/2014 8:37 PM	META File	1 KB
gamePlay.meta	12/8/2014 8:37 PM	META File	1 KB
highWayGameplay.meta	12/8/2014 8:37 PM	META File	1 KB
highWayGameplay.unity	3/31/2015 8:24 PM	Unity scene file	1,203 KB
highWayGameplay.unity.meta	12/8/2014 8:35 PM	META File	1 KB
MainMenu.unity	3/27/2015 3:30 AM	Unity scene file	1,314 KB
MainMenu.unity.meta	12/8/2014 8:37 PM	META File	1 KB
splashScreen.unity	12/8/2014 8:37 PM	Unity scene file	8 KB
splashScreen.unity.meta	12/8/2014 8:37 PM	META File	1 KB

Figure 4.2: Hierarchy and Scene

There was 5 main scenes of this project. Then, we open highWaysGamePlay scene by unity programmer tool:

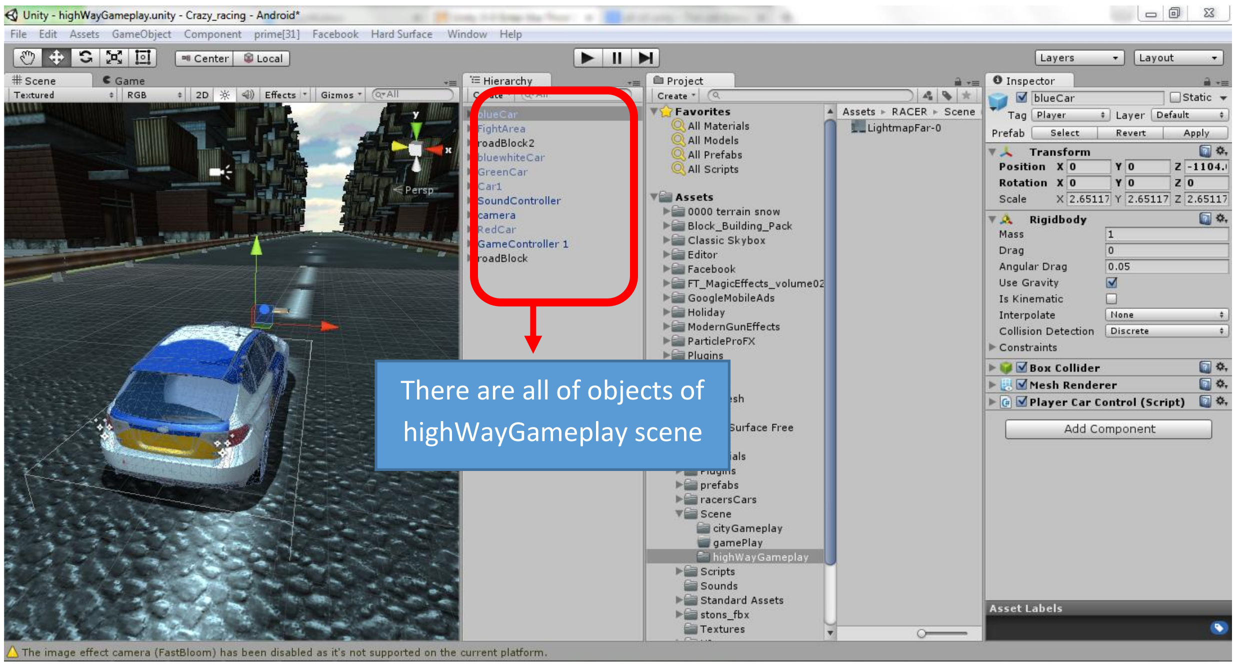


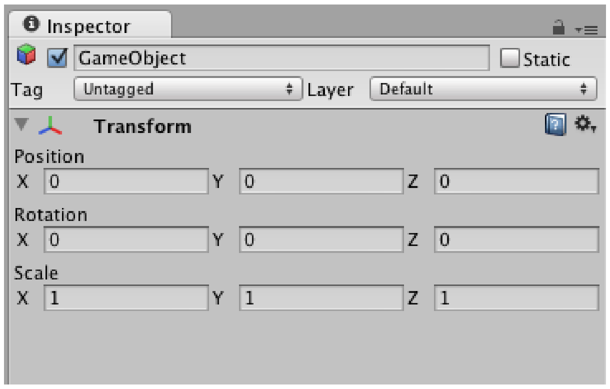
Figure 4.3: Game objects of a scene

So, what is game objects in Unity3d.

1.2 Game Object

Any active object in the currently open scene is called a Game Object. Certain assets taken from the Project panel such as models and prefabs become game objects when placed (or 'instantiated') into the current scene. Other objects such as particle systems and primitives can be placed into the scene by using the Create button on the Hierarchy or by using the GameObjects menu at the top of the interface. All GameObjects contain at least one component to begin with, that is, the Transform component. Transform simply tells the Unity engine the position, rotation, and scale of an object—all described in X, Y, Z coordinate (or in the case of scale, dimensional) order. In turn, the component can then be addressed in scripting in order to set an object's position, rotation, or scale. From this initial component, you will build upon GameObjects with further components, adding required functionality to build every part of any game scenario you can imagine.

In the following image, you can see the most basic form of a Game Object, as shown in the Inspector panel:



Inspector

GameObject ☐ Static

Tag: Untagged | Layer: Default

Transform

Position X 0 Y 0 Z 0

Rotation X 0 Y 0 Z 0

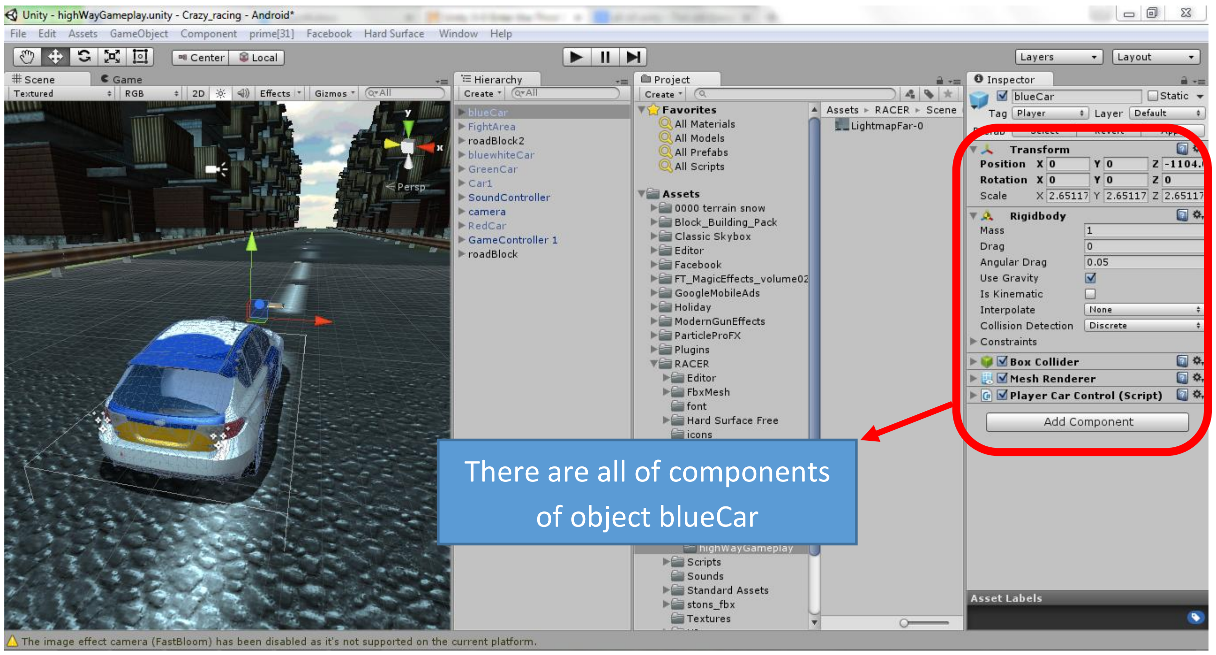
Scale X 1 Y 1 Z 1

Figure 4.4: Transform

Game Objects can also be nested in the Hierarchy, in order to create the parent-child relationships mentioned previously.

Next, we go to game object's component.

1.3 Component



There are all of components of object blueCar

Inspector

blueCar ☐ Static

Tag: Player | Layer: Default

Transform

Position X 0 Y 0 Z -1104.

Rotation X 0 Y 0 Z 0

Scale X 2.65117 Y 2.65117 Z 2.65117

Rigidbody

Mass 1

Drag 0

Angular Drag 0.05

Use Gravity ✓

Is Kinematic ☐

Interpolate None

Collision Detection Discrete

Constraints

✓ Box Collider

✓ Mesh Renderer

✓ Player Car Control (Script)

Add Component

The image effect camera (FastBloom) has been disabled as it's not supported on the current platform.

Figure 4.5: Components of a object

Components come in various forms. They can be for creating behavior, defining appearance, and influencing other aspects of an object's function in the game. By attaching components to an object, you can immediately apply new parts of the game engine to your object. Common components of game production come built-in with Unity, such as the Rigidbody component mentioned earlier, down to simpler elements such as lights, cameras, particle emitters, and more. To build further interactive elements of the game, you'll write scripts, which are also treated as components in

Unity. Try to think of a script as something that extends or modifies the existing functionality available in Unity or creates behavior with the Unity scripting classes provided.

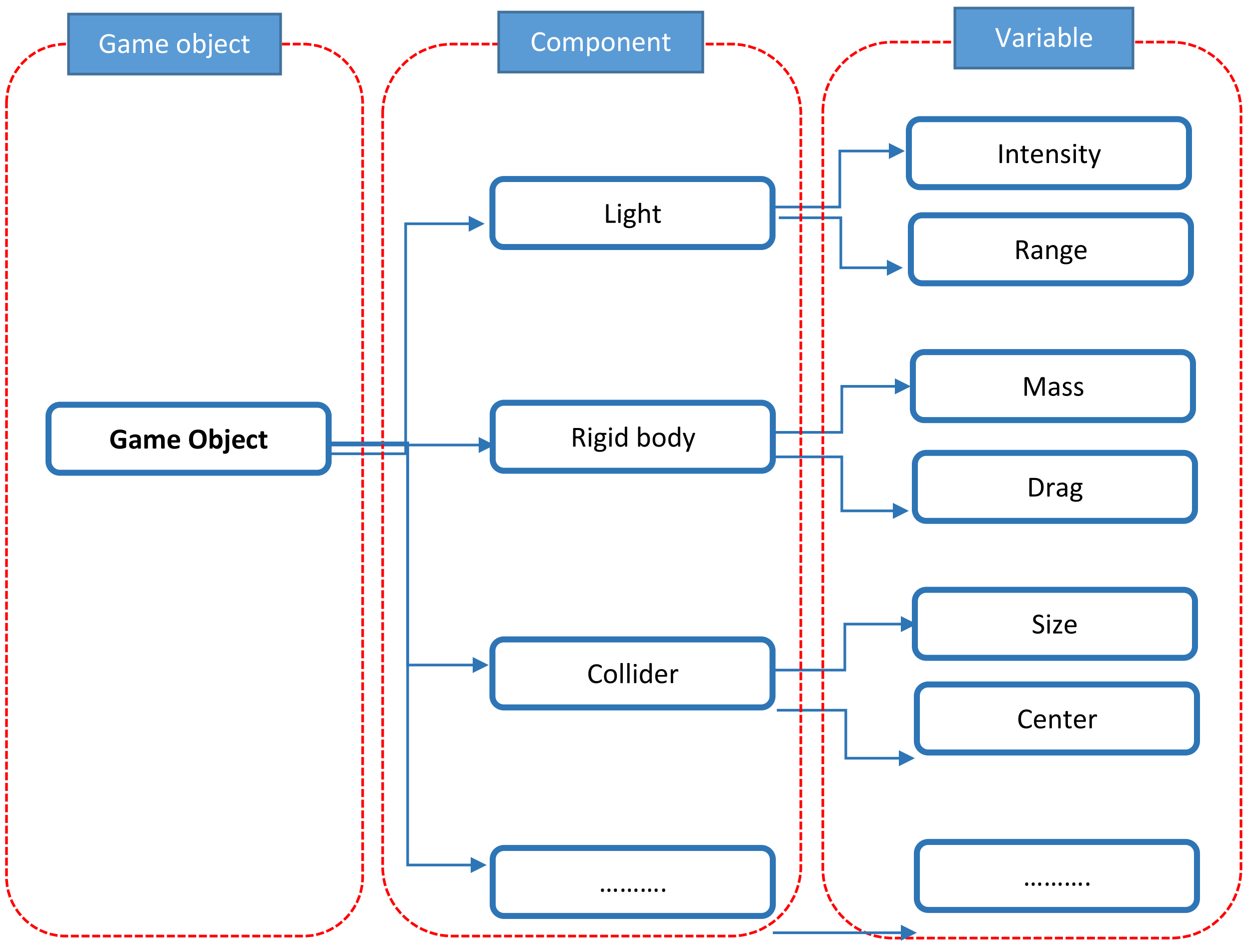


Figure 4.6: Hierarchy of a game object

In real, game object may contain more than 3 components above, and each component contains less or more than 2 variables like that.

Firstly, we should take a look to some popular component:

1.3.1 Rigidbody

In game engines, there is no assumption that an object should be affected by physics—firstly because it requires a lot of processing power, and secondly because there is simply no need to do so. For example, in a 3D driving game, it makes sense for the cars to be under the influence of the physics engine, but not the track or surrounding objects, such as trees, walls, and so on—they will remain static for the duration of the game. For this reason, when making games in Unity a Rigidbody physics component is given to any object that you wish to be under the control of the physics engine, and ideally any moving object, so that the physics engine is aware of the moving object, to save on performance.

Physics engines for games use the Rigidbody dynamics system of creating realistic motion. This simply means that instead of objects being static in the 3D world, they can have properties such as mass, gravity, velocity, and friction.

1.3.2 Collision detection

More crucial in game engines than in 3D animation, collision detection is the way we analyze our 3D world for inter-object collisions. By giving an object a Collider component, we are effectively placing an invisible net around it. This net usually mimics its shape and is in charge of reporting any collisions with other colliders, making the game engine respond accordingly.

There are two main types of Collider in Unity—Primitives and Meshes. Primitive shapes in 3D terms are simple geometric objects such as Boxes, Spheres, and Capsules. Therefore, a primitive collider such as a Box collider in Unity has that shape, regardless of the visual shape of the 3D object it is applied to. Often, Primitive colliders are used because they are computationally cheaper or because there is no need for precision. A Mesh collider is more expensive as it can be based upon the shape of the 3D mesh it is applied to; therefore, the more complex the mesh, the more detailed and precise the collider will be, and more computationally expensive it will become. However, as shown in the Car tutorial example earlier, it is possible to assign a simpler mesh than that which is rendered, in order to create simpler and more efficient mesh colliders.

1.3.3 Mesh Renderer

Meshes imported from 3D packages can use multiple Materials. All the materials used by a Mesh Renderer are held in the Materials list. Each submesh will use one material from the materials list. If there are more materials assigned to the Mesh Renderer than there are submeshes in the mesh, the first submesh will be rendered with each of the remaining materials, one on top of the next. At a cost of performance, this will let you set up multi-pass rendering on that submesh. Fully opaque materials, however, will simply overwrite the previous layers, costing performance for no advantage.

1.3.4 Scripting

Scripting is an essential ingredient in all games. Even the simplest game will need scripts to respond to input from the player and arrange for events in the gameplay to happen when they should. Beyond that, scripts can be used to create graphical effects, control the physical behavior of objects or even implement a custom AI system for characters in the game.

Scripting is a skill that takes some time and effort to learn; the intention of this section is not to teach you how to write script code from scratch but rather to explain the main concepts that apply to scripting in Unity.

1.3.5 Materials, textures, shaders

Materials are a common concept to all 3D applications, as they provide the means to set the visual appearance of a 3D model. From basic colors to reflective image-based surfaces, materials handle everything.

Let's start with a simple color and the option of using one or more images—known as textures. In a single material, the material works with the shader, which is a script in charge of the style of rendering. For example, in a reflective shader, the material will render reflections of surrounding objects, but maintain its color or the look of the image applied as its texture.

In Unity, the use of materials is easy. Any materials created in your 3D modeling package will be imported and recreated automatically by the engine and created as assets that are reusable. You can also create your own materials from scratch, assigning images as textures and selecting a shader from a large library that comes built-in. You may also write your own shader scripts or copy-paste those written by fellow developers in the Unity community, giving you more freedom for expansion beyond the included set.

When creating textures for a game in a graphics package such as Photoshop or GIMP, you must be aware of the resolution. Larger textures will give you the chance to add more detail to your textured models, but be more intensive to render. Game textures imported into Unity will be scaled to a power of 2 resolution. For example:

- 64px x 64px

- 128px x 128px

- 256px x 256px

- 512px x 512px

- 1024px x 1024px

Creating textures of these sizes with content that matches at the edges will mean that they can be tiled successfully by Unity. You may also use textures scaled to values that are not powers of two, but mostly these are used for GUI elements.

So, we will take a look to our project:

- The CR project is a game as user view, each device run this game as a client/server to play with the others.

- There are 2 main scenes in CR project: main menu and game play

+ Main menu:

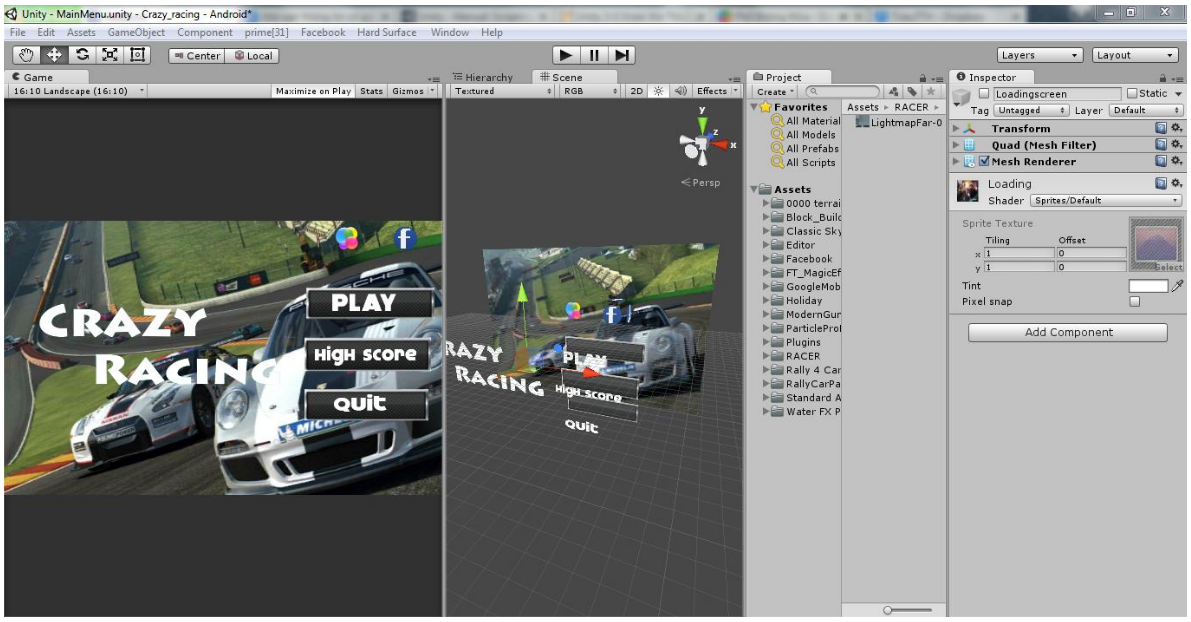


Figure 4.7: Main menu view in Unity tools

MainMenu		
Objects	**Components**	**Descriptions**
LoadingScreen	- Transform - Quad (Mesh Filter) - Mesh Renderer - Shader (Loading picture)	In coming scene of this app.

SoundController	- Transform - SoundController script - Audio - TakeScreen Shot script	Manage all of sounds in this scene.
UIContainer	- Transform	Manage all of UI Texts – Labels.
MainCamera	- Transform - Camera - GUILayer - FlareLayer - AudioListener - MouseOrbit	Main view to this scene.
ShowCase	- Transform - Cycle (Mesh Filter) - Mesh Renderer - Animator - Shader	Under the car, it will be used to show car scene.
AdMod	- Google Mobile Ads Script - Ads banner Script - Ad Mod Interitial Script	Used to add advertisements.

+ Game play:

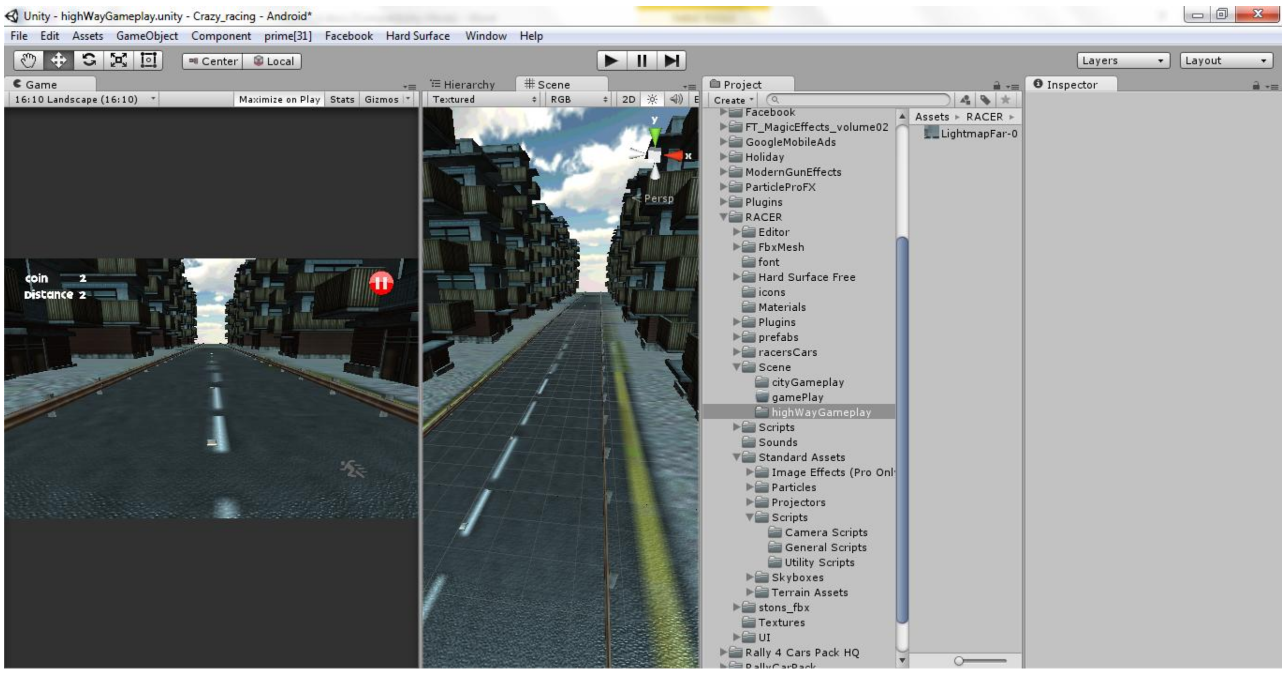


Figure 4.8: Game play view in Unity tools

Gameplay		
Objects	**Components**	**Descriptions**
SoundController	- Transform - SoundController Script - Audio Source - Take Screen Shot Script	Manage all of sounds in this scene.
GameController	- Transform - GameController Script	Manage all flows/actions in this scene.
MainCamera	- Transform - Camera - GUILayer	Main view to this scene.

	- FlareLayer - AudioListener	
Road block 1	- Transform - Cube (Mesh Filter) - Box Collider - RoadGenerator Script	First road block – street.
Road block 2	- Transform - Cube (Mesh Filter) - Box Collider - RoadGenerator Script	Second road block – this street will be pawned after the 1st road block is travelled.
Car	- Transform - Rigidbody - Box Collider - Mesh Renderer - PlayerCarControl Script	Player's car.

1.4 Network View technology

To help developer create a game that is able to multi play, Unity introduce to Network View technology. Network Views are the gateway to creating networked multiplayer games in Unity.

Network Views keep watch on particular objects to detect changes. These changes are then shared to the other clients on the network to ensure the change of state is noted by all of them.

There are the main component involved in sharing data across the network. They allow two kinds of network communication: State Synchronization and Remote Procedure Calls.

+ Remote Procedure Calls(RPC): Remote Procedure Calls (RPCs) let you call functions on a remote machine. Invoking an RPC is similar to calling a normal function and almost as easy but there are some important differences to understand.

+State Synchronization: It’s reponisbility is synchronize Transform, Animation, Rigidbody and MonoBehaviour components.

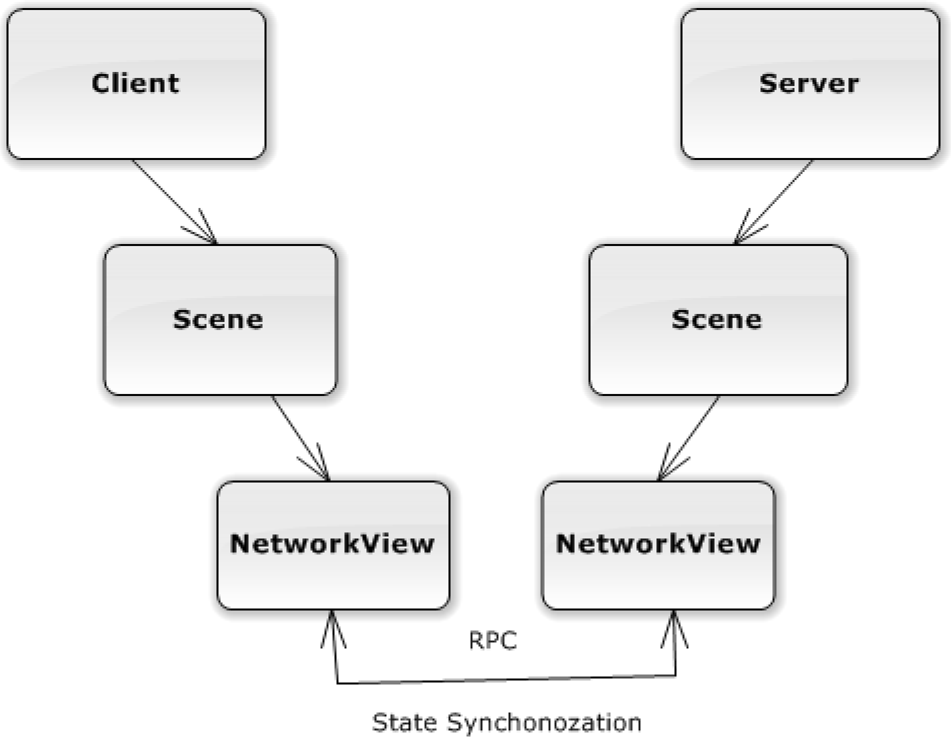


Figure 4.9: Overall multiplayer

2. Crazy Racing's structure

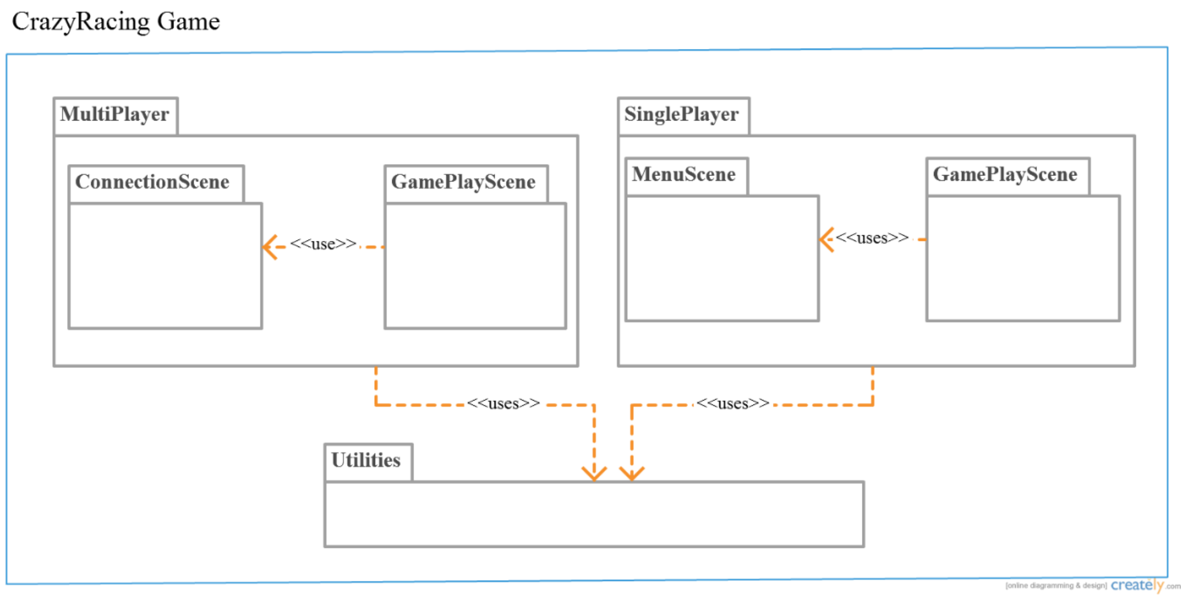


Figure 4.10: Crazy Racing's structure

Crazy Racing game provides two mode: Single Player and Multiplayer. Each mode have two scene as you can see in diagram above. Beside, utilities package is used to help some common game's function.

2.1 SinglePlayer

2.1.1 MenuScene

The reposibility of this scene is:

+ Displays four main options when game start: Play, MultiPlayer, HighScore, Quit
+ Allows user choice: Map, car, sound
+ Allows user unlock and buy item such as: Map, Car, Sound.

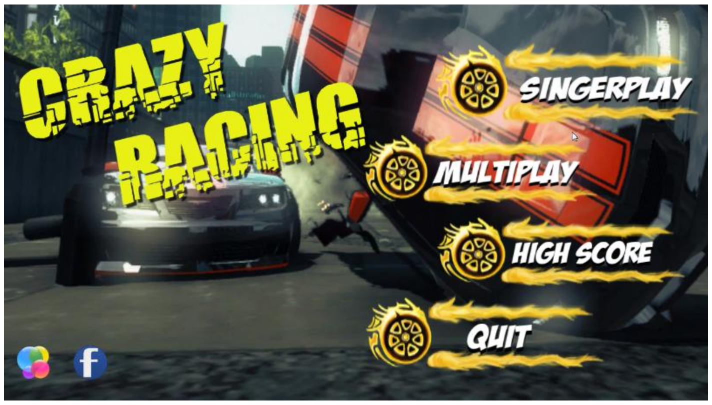


Figure 4.11: Main menu

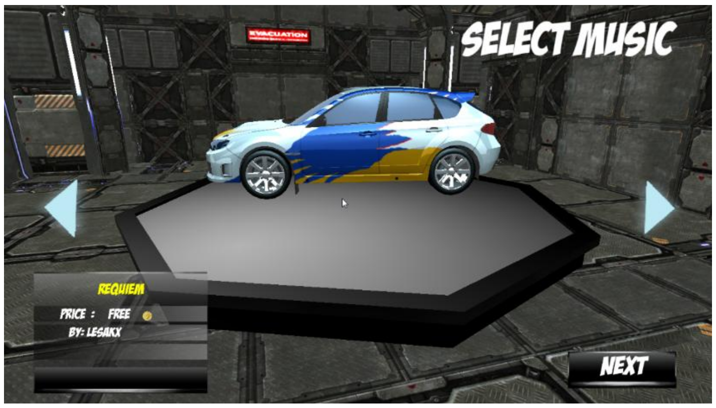


Figure 4.12: Choose music/car

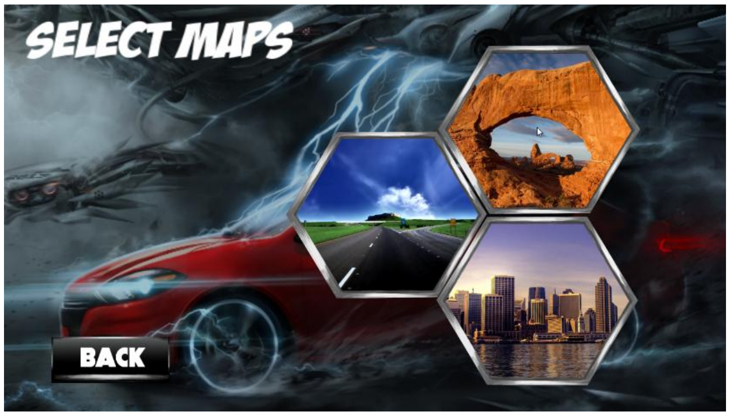


Figure 4.13: Select maps

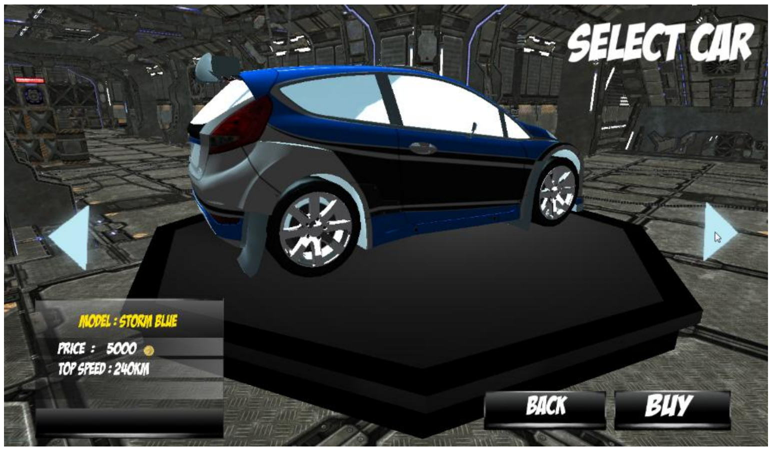


Figure 4.14: Select car

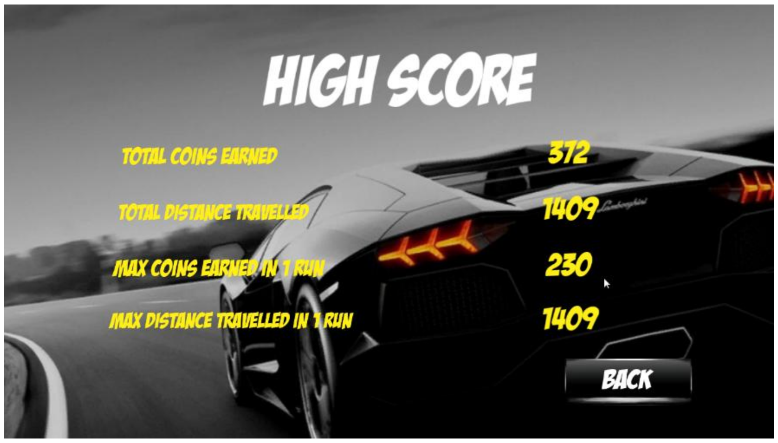


Figure 4.14: Highscore

2.1.2 GamePlayScene

The reposibility of this scene is: Displays object in game: such as: Car, Roadm Building, Power up item, Tree, Particle Object(Smoke, Fire, Power up item amination), Helicopter, allows user controls the car, manage power up item, helicopter, road, camera, building.

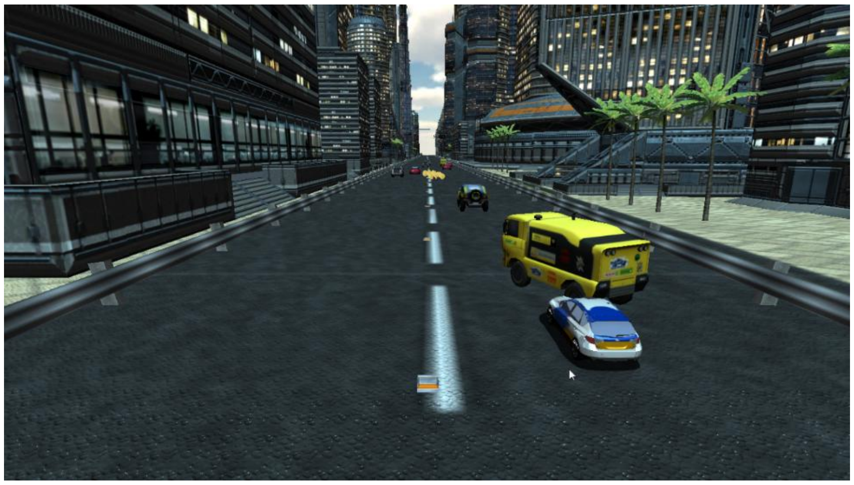


Figure 4.15: Gameplay 1

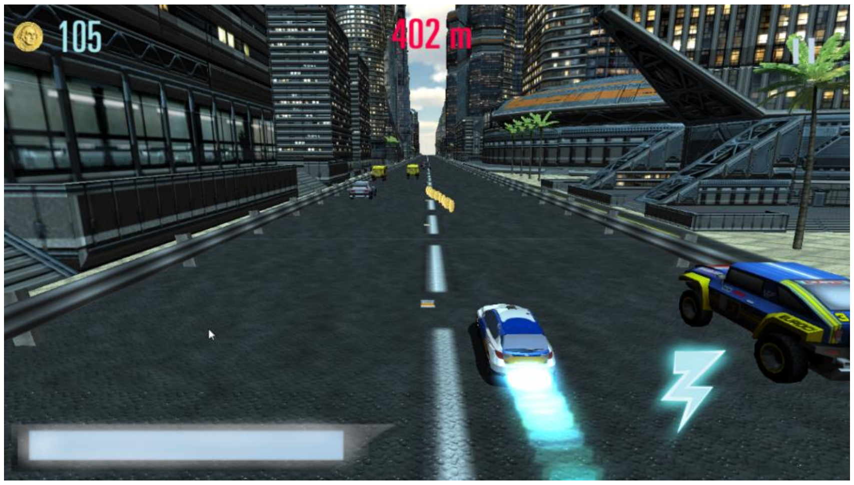


Figure 4.16: Gameplay 2

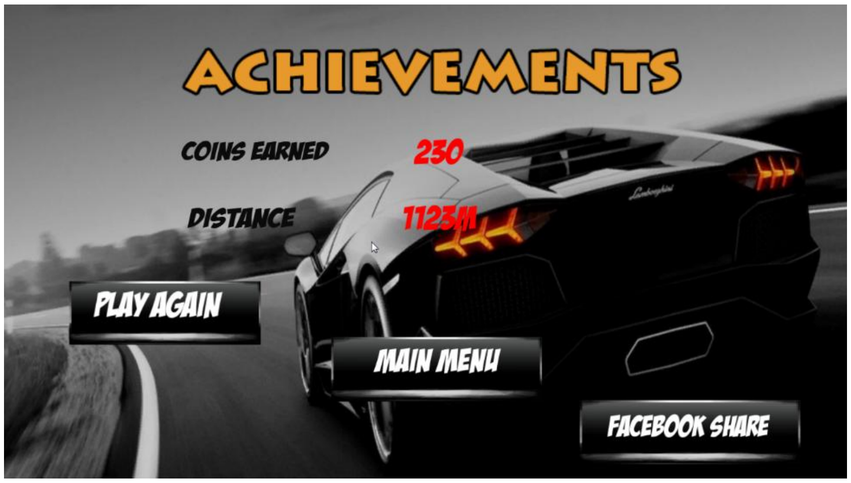
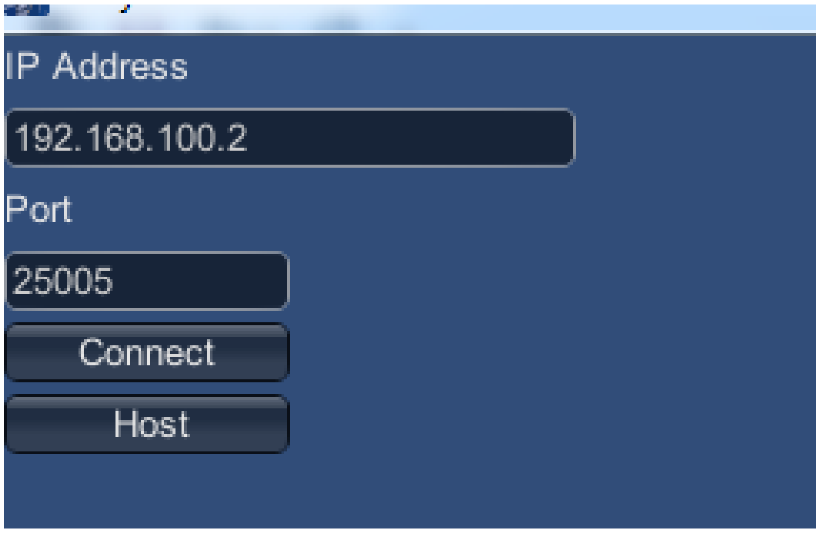


Figure 4.17: Gameover

2.2 MultiPlayer

2.2.1 ConnectionScene

The reposibility of this scene is displays available player who want to play game in multiplayer mode, allows user choice competitor.


IP Address

192.168.100.2

Port

25005

Figure 4.18: Multiplayer connect

2.2.2 GamePlayScene

The reposibility of this scene is:allow two users play together synchronously, displays object in game: such as: Car, Roadm Building, Power up item, Tree, Particle Object(Smoke, Fire, Power up item amination), Helicopter, allows user controls the car, manage power up item, helicopter, road, camera, building.

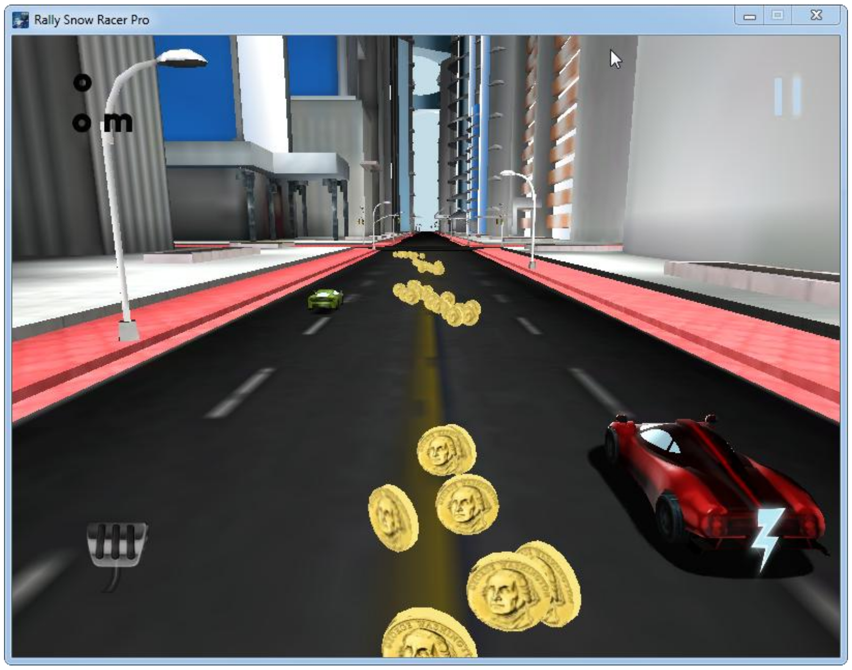
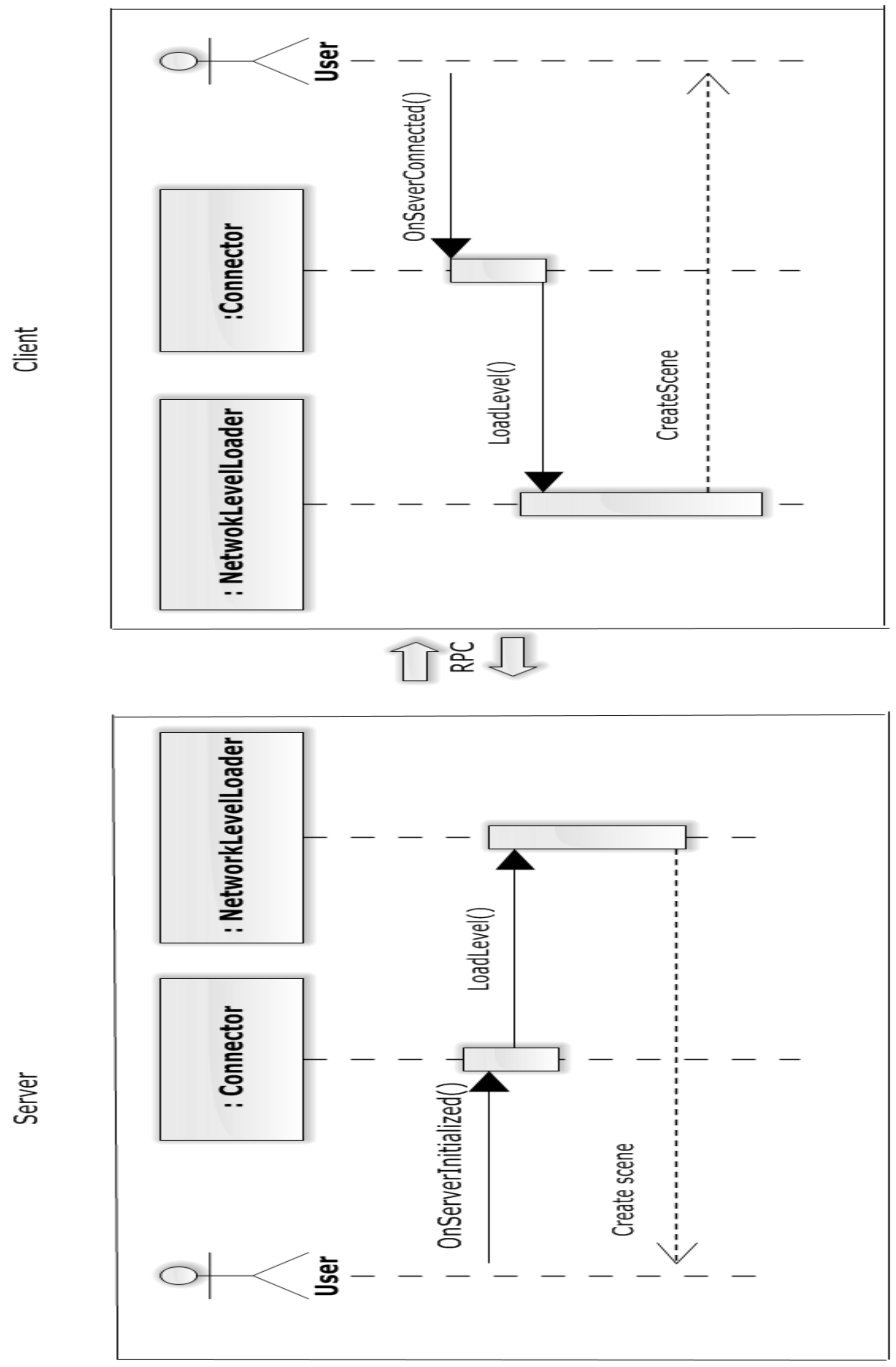


Figure 4.19: Game play multiplayer

2.2.3 How Multiplayer works in our game

Crazy Racing game use peer to peer protocol with supported Unity Network View to imlelents multiplayer game mode



2.3 Utilities

This package is written for managing database in game, share achivement and support for game objecs of scene. We will descripts more in III. Detais Description.

III. Details Description of Components

1. SinglePlayer

MenuScene

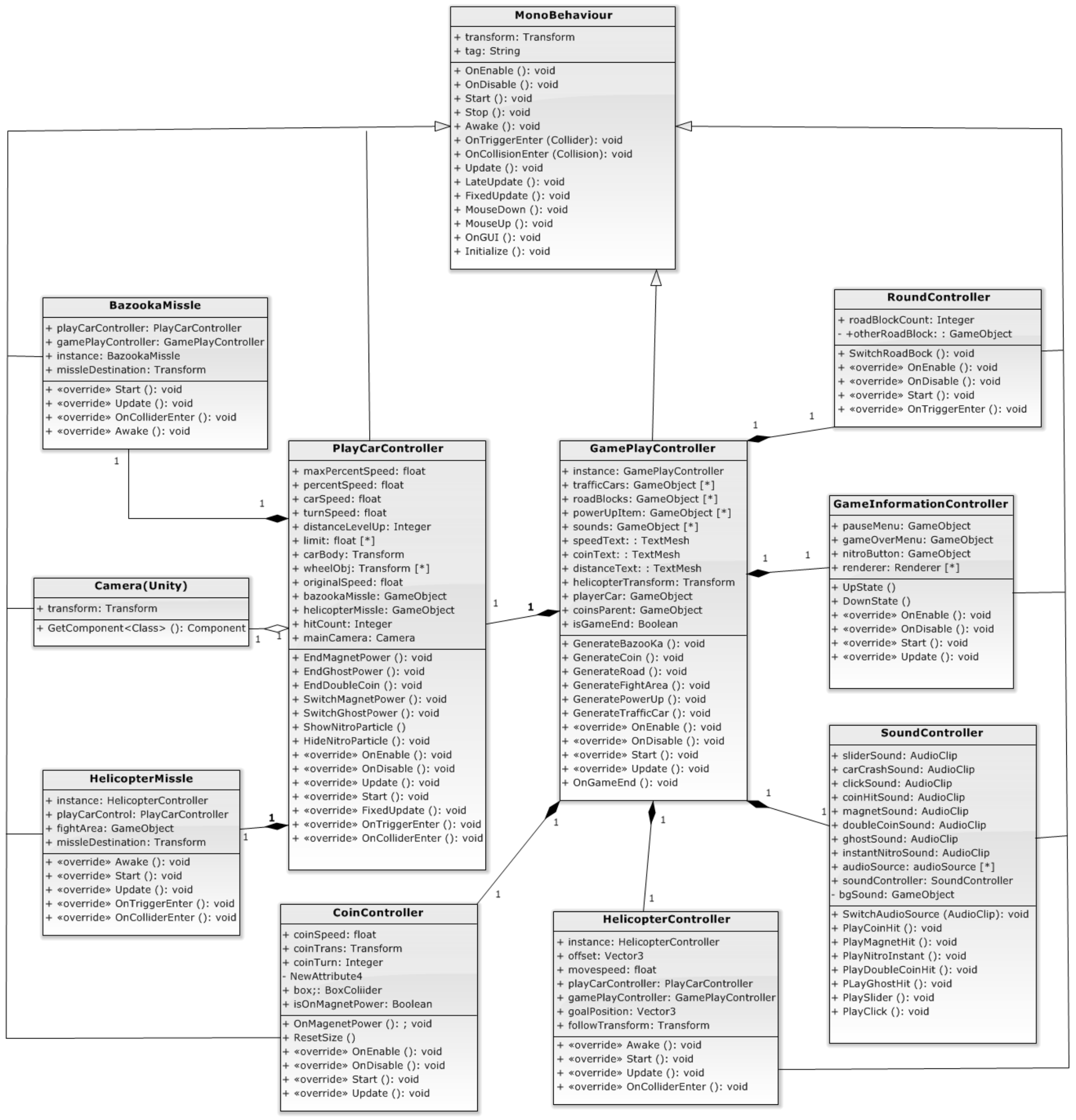

MonoBehaviour
+ transform: Transform
+ tag: String
+ OnEnable (): void
+ OnDisable (): void
+ Start (): void
+ Stop (): void
+ Awake (): void
+ OnTriggerEnter (Collider): void
+ OnCollisionEnter (Collision): void
+ Update (): void
+ LateUpdate (): void
+ FixedUpdate (): void
+ MouseDown (): void
+ MouseUp (): void
+ OnGUI (): void
+ Initialize (): void
BazookaMissle
+ playCarController: PlayCarController
+ gamePlayController: GamePlayController
+ instance: BazookaMissle
+ missleDestination: Transform
+ «override» Start (): void
+ «override» Update (): void
+ «override» OnColliderEnter (): void
+ «override» Awake (): void
RoundController
+ roadBlockCount: Integer
- +otherRoadBlock: : GameObject
+ SwitchRoadBock (): void
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Start (): void
+ «override» OnTriggerEnter (): void
PlayCarController
+ maxPercentSpeed: float
+ percentSpeed: float
+ carSpeed: float
+ turnSpeed: float
+ distanceLevelUp: Integer
+ limit: float [*]
+ carBody: Transform
+ wheelObj: Transform [*]
+ originalSpeed: float
+ bazookaMissle: GameObject
+ helicopterMissle: GameObject
+ hitCount: Integer
+ mainCamera: Camera
+ EndMagnetPower (): void
+ EndGhostPower (): void
+ EndDoubleCoin (): void
+ SwitchMagnetPower (): void
+ SwitchGhostPower (): void
+ ShowNitroParticle ()
+ HideNitroParticle (): void
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Update (): void
+ «override» Start (): void
+ «override» FixedUpdate (): void
+ «override» OnTriggerEnter (): void
+ «override» OnColliderEnter (): void
GamePlayController
+ instance: GamePlayController
+ trafficCars: GameObject [*]
+ roadBlocks: GameObject [*]
+ powerUpItem: GameObject [*]
+ sounds: GameObject [*]
+ speedText: : TextMesh
+ coinText: : TextMesh
+ distanceText: : TextMesh
+ helicopterTransform: Transform
+ playerCar: GameObject
+ coinsParent: GameObject
+ isGameEnd: Boolean
+ GenerateBazooKa (): void
+ GenerateCoin (): void
+ GenerateRoad (): void
+ GenerateFightArea (): void
+ GeneratePowerUp (): void
+ GenerateTrafficCar (): void
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Start (): void
+ «override» Update (): void
+ OnGameEnd (): void
GameInformationController
+ pauseMenu: GameObject
+ gameOverMenu: GameObject
+ nitroButton: GameObject
+ renderer: Renderer [*]
+ UpState ()
+ DownState ()
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Start (): void
+ «override» Update (): void
Camera(Unity)
+ transform: Transform
+ GetComponent<Class> (): Component
HelicopterMissle
+ instance: HelicopterController
+ playCarControl: PlayCarController
+ fightArea: GameObject
+ missleDestination: Transform
+ «override» Awake (): void
+ «override» Start (): void
+ «override» Update (): void
+ «override» OnTriggerEnter (): void
+ «override» OnColliderEnter (): void
SoundController
+ sliderSound: AudioClip
+ carCrashSound: AudioClip
+ clickSound: AudioClip
+ coinHitSound: AudioClip
+ magnetSound: AudioClip
+ doubleCoinSound: AudioClip
+ ghostSound: AudioClip
+ instantNitroSound: AudioClip
+ audioSource: audioSource [*]
+ soundController: SoundController
- bgSound: GameObject
+ SwitchAudioSource (AudioClip): void
+ PlayCoinHit (): void
+ PlayMagnetHit (): void
+ PlayNitroInstant (): void
+ PlayDoubleCoinHit (): void
+ PLayGhostHit (): void
+ PlaySlider (): void
+ PlayClick (): void
CoinController
+ coinSpeed: float
+ coinTrans: Transform
+ coinTurn: Integer
- NewAttribute4
+ box;: BoxColiider
+ isOnMagnetPower: Boolean
+ OnMagenetPower (): ; void
+ ResetSize ()
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Start (): void
+ «override» Update (): void
HelicopterController
+ instance: HelicopterController
+ offset: Vector3
+ movespeed: float
+ playCarController: PlayCarController
+ gamePlayController: GamePlayController
+ goalPosition: Vector3
+ followTransform: Transform
+ «override» Awake (): void
+ «override» Start (): void
+ «override» Update (): void
+ «override» OnColliderEnter (): void
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1

Figure 4.20: Menuscreen

1.1 BuyPopUP

Properties

Name	Type	Visibility	Description
costText	TextMesh	public	Display Car’s Cost
carSelectionMenu	GameObject	public	Car selection menu
uiCamera	Camera	public	User interface camera
carCost	int	public static	Cost of car

Operations

Signature: void OnEnable()

Description: Register event

Parameter’s name	Type	Description
N/A		

Signature: void Start()

Description:

Parameter’s name	Type	Description
N/A		

Signature: void SetInt ()

Description: Sets the value of the preference identified by key.

Parameter’s name	Type	Description
carIndex		

Signature: void Update()

Description: Update mouse position		
Parameter's name	**Type**	**Description**
N/A		

Signature: void MouseUp () **Description**: Car selection and purchase		
Parameter's name	**Type**	**Description**
a	Vector3	

1.2 BuyPopUPSound

Properties

Name	**Type**	**Visibility**	**Description**
costText	TextMesh	public	Display Car's Cost
soundTrackMenu	GameObject	public	Sound
uiCamera	Camera	public	User interface camera
soundTrackCost	int	public static	Cost of sound track

Operations

Signature: void OnEnable() **Description**: Register event		
Parameter's name	**Type**	**Description**
N/A		

Signature: void Start()   **Description**:

Parameter's name	**Type**	**Description**
N/A		
Signature: void Update() **Description**: Update mouse position		
Parameter's name	**Type**	**Description**
N/A		

Signature: void MouseUp (Vector3 position) **Description**: Soundtrack selection and purchase		
Parameter's name	**Type**	**Description**
position	Vector3	User's touch position

1.3 CarSelection

Properties

Name	**Type**	**Visibility**	**Description**
uiCamera	Camera	public	User interface camera
buttonRenders	Renderer[]	public	Collection of button renderer
buttonTexture	Texture[]	public	Texture button
hit	RaycastHit	public	Raycast hit
buyButton	GameObject	public	Buy button
playButton	GameObject	public	Play button
buyPopUp	GameObject	public	Buy popup
InAPPMenu	GameObject	public	In app purchase menu
InAPPMenuAndroid	GameObject	public	Android in app purchase menu

loadingLevelObj	GameObject	public	Level load object
menuObj	GameObject	public	Menu
IAP	GameObject	public	
carIndex	int	public static	Car's index
carMeshObjs	GameObject[]	public	
carSpeedDisplayText	TextMesh	public	Display car's speed
carPriceDisplayText	TextMesh	public	Display car's price
headingText	TextMesh	public	Heading text

Operations

Signature: void Start()

Description:

Parameter's name	**Type**	**Description**
N/A		

Signature: void Update()

Description:

Parameter's name	**Type**	**Description**
N/A		

Signature: void MouseUp (Vector3 position)

Description: mouse position control

Parameter's name	**Type**	**Description**
position	Vector3	Get user's touch position

Signature: void MouseDown(Vector3 position) **Description**: mouse position control		
Parameter's name	**Type**	**Description**
position	Vector3	Get user's touch position

Signature: void ShowNextcar () **Description**: show next car		
Parameter's name	**Type**	**Description**
N/A		

Signature: void ShowPreviouscar () **Description**: show previous car		
Parameter's name	**Type**	**Description**
N/A		

Signature: void ShowCar () **Description**: show car		
Parameter's name	**Type**	**Description**
N/A		

Signature: void OnEnable () **Description**: Register event for playing or buying car		
Parameter's name	**Type**	**Description**
N/A		

Signature: void ShowcarINFO () **Description**: show car information		
Parameter's name	**Type**	**Description**
N/A		

Signature: void Purchasecars () **Description**: purchase car selected		
Parameter's name	**Type**	**Description**
N/A		

1.4 EndScoreDisplayer:

Properties

Name	**Type**	**Visibility**	**Description**
distancetext	TextMesh	public	Display player's distance
coinsText	TextMesh	public	Display coin that player gained
playAgainButton	GameObject	public	Play again button
mainMenuButton	GameObject	public	Back main menu button
facebookShare	GameObject	public	Facebook share button
originalPositions	Vector3[]	public	Original Position
showFullScreenAd	event	public static	Advertising screen

Operations

Signature: void Start()   **Description**: Give value of score after gaming

Parameter’s name	Type	Description
N/A		

Signature: ReportScore() **Description**: Report a score to a specific leaderboard.		
Parameter’s name	**Type**	**Description**
distanceTravelled, DistanceID	long, string	
collectedCoinsCounts, CoinsID	long, string	

Signature: void OnEnable() **Description**: Register event		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void SetInt () **Description**: Sets the value of the preference identified by key.		
Parameter’s name	**Type**	**Description**
TotalCoins, collectedCoinsCounts	string, int	
TotalCoinsHighscore, collectedCoinsCounts	string, int	
MaxCoins, collectedCoinsCounts	string, int	
TotalDistanceHighscore, distanceTravelled	string, int	
MaxDistance, distanceTravelled	string, int	

Signature: GetInt () **Description**: get number of total, hight score and max coins		
Parameter's name	**Type**	**Description**
TotalCoins	int	Total coin
TotalCoinsHighscore	int	Totai coin in highscore data
MaxCoins	int	Max coin
TotalDistanceHighscore	int	Totai distance in highscore data
MaxDistance	int	Max distance

Signature: void ChangeCoinText () **Description**: Display number of coins that player gained		
Parameter's name	**Type**	**Description**
newValue	float	The current coin got.

Signature: void StartDistanceCount () **Description**: Count distance and record		
Parameter's name	**Type**	**Description**
N/A		

Signature: PlayCoinHit() **Description**		
Parameter's name	**Type**	**Description**

N/A		

Signature: PlayCarCrashSound () **Description**		
Parameter's name	**Type**	**Description**
N/A		

Signature: PlayPowerPickUp() **Description**		
Parameter's name	**Type**	**Description**
N/A		

Signature: void ChangeDistanceText () **Description**: Display number of distance that player gained		
Parameter's name	**Type**	**Description**
newValue	float	

Signature: void ShowButtons () **Description**: show button after racing		
Parameter's name	**Type**	**Description**
N/A		

Signature: PlaySlider()   **Description**:

Parameter’s name	Type	Description
N/A		

1.5 freeCoinsADPromotion:

Properties

Name	Type	Visibility	Description
showCointainer	GameObject	public	Show container that should be displayed
notAvailContainer	GameObject	public	Hide container
carSelectionMenu	GameObject	public	Car selection menu
uiCamera	Camera	public	User interface camera
alreadyGiveFreeCoins	bool	public static	

Operations

Signature: void OnEnable()

Description: Initize variable.

Parameter’s name	Type	Description
N/A		

Signature: void Update ()

Description: Update mouse position

Parameter’s name	Type	Description
N/A		

Signature: void MouseUp (Vector3 position)

Description: mouse position control		
Parameter's name	**Type**	**Description**
position	Vector3	User's touch position

Signature: void ScreenPointToRay (Vector3 position) **Description**: Returns a ray going from camera through a screen point.		
Parameter's name	**Type**	**Description**
position	Vector3	User's touch position

Signature: void PlayClickSound () **Description**:		
Parameter's name	**Type**	**Description**
N/A		

1.6 FreeCoinsADpromotionSound:

Properties

Name	**Type**	**Visibility**	**Description**
showCointainer	GameObject	public	
notAvailContainer	GameObject	public	
soundSelectionMenu	GameObject	public	Sound selection menu
uiCamera	Camera	public	User interface camera
alreadyGiveFreeCoins	bool	public static	

Operations

Signature: void Update ()		
Description: update mouse position		
Parameter's name	**Type**	**Description**
N/A		

Signature: void MouseUp ()		
Description: mouse position control		
Parameter's name	**Type**	**Description**
a	Vector3	

Signature: void PlayClickSound ()		
Description:		
Parameter's name	**Type**	**Description**
N/A		

Signature: void SetActive ()		
Description: Activates/Deactivates the GameObject.		
Parameter's name	**Type**	**Description**
	bool	

1.7 GameCenterUI

Properties

Name	**Type**	**Visibility**	**Description**
N/A			

Operations

Signature: void Start() **Description**: Initialize game center		
Parameter's name	**Type**	**Description**
N/A		

Signature: void Update () **Description**:		
Parameter's name	**Type**	**Description**
N/A		

1.8 HighScoreMenu

Properties

Name	**Type**	**Visibility**	**Description**
uiCamera	Camera	public	User interface camera
coinsText	TextMesh	public	Displays coins that player gain.
distancetext	TextMesh	public	Display player's distance
maxCoinsText	TextMesh	public	Display record of coins
maxDistanceText	TextMesh	public	Display record of distance
coins	float	private	Count coins variable
distance	float	private	Count distance variable
maxCoins	float	private	Max coins variable
maxDistance	float	private	Max distance variable
mainMenuButton	GameObject	public	Back to main menu button

menuButtonRenders	Renderer[]	public	Collection of menu button renderer
buttonTexture	Texture[]	public	buttonTexture
hit	RaycastHit	public	Raycast hit
isDrag	bool		

Operations

Signature: void Update ()

Description: Update mouse position

Parameter's name	**Type**	**Description**
N/A		

Signature: void OnGUI ()

Description:

Parameter's name	**Type**	**Description**
N/A		

Signature: void MouseUp ()

Description: Mouse position control

Parameter's name	**Type**	**Description**
position	Vector3	User's touch position

Signature: void ScreenPointToRay (Vector3 position)

Description: Returns a ray going from camera through a screen point.

Parameter's name	**Type**	**Description**

position	Vector3	User’s touch position

Signature: void MouseDown () **Description**: Mouse position control		
Parameter’s name	**Type**	**Description**
a	Vector3	User’s touch position

Signature: void OnEnable () **Description**: Give value class’s properties		
Parameter’s name	**Type**	**Description**
N/A		

1.9 LevelSelection

Properties

Name	**Type**	**Visibility**	**Description**
hit	RaycastHit	public	Raycast hit
uiCamera	Camera	public	User interface camera
levelName	string	public static	Name of level
soundTrackSelection	GameObject	public	Select soundtrack
LadingSpin	GameObject	public	
buttonRenders	Renderer	public	Button render
ButtonTexture	Texture[]	public	Button textture

Operations

Signature: void Start () **Description**:		
Parameter's name	**Type**	**Description**
N/A		

Signature: void Update () **Description**: Update mouse position		
Parameter's name	**Type**	**Description**
N/A		

Signature: void MouseUp (Vector 3 position) **Description**: Mouse position control		
Parameter's name	**Type**	**Description**
position	Vector3	User's touch position

Signature: void ScreenPointToRay (Vector 3 position)) **Description**: Returns a ray going from camera through a screen point.		
Parameter's name	**Type**	**Description**
position	Vector3	User's touch position

Signature: void MouseUp (Vector 3 position) **Description**: Mouse position control		
Parameter's name	**Type**	**Description**
position	Vector3	User's touch position

Signature: void MouseDown (Vector 3 position) **Description**: Mouse position control		
Parameter's name	**Type**	**Description**
position	Vector3	User's touch position

1.10 MainMenu

Properties

Name	**Type**	**Visibility**	**Description**
gameCenter	GameObject	public	Game center object
uiCamera	Camera	public	User interface camera
menuButtonRenders	Renderer[]	public	Menu button renders
buttonTexture	Texture[]	public	Button texture
hit	RaycastHit	public	Raycast hit
storeObject	GameObject	public	Store object
carSelection	GameObject	public	Car selection object
highScore	GameObject	public	High Score
reviewUrls	string[]	public	Web address to review game website
MoreUrls	string[]	public	Some useful web address
isDrag	bool		

Operations

Signature: void Start () **Description**:		
Parameter's name	**Type**	**Description**
N/A		

Signature: void Update () **Description**: Update mouse position		
Parameter's name	**Type**	**Description**
N/A		

Signature: void OnGUI () **Description**: OnGUI is called for rendering and handling GUI events		
Parameter's name	**Type**	**Description**
N/A		

Signature: void MouseDown (Vector3 positon) **Description**: mouse position control		
Parameter's name	**Type**	**Description**
position	Vector3	User's touch event

Signature: void ScreenPointToRay (Vector3 position) **Description**: Returns a ray going from camera through a screen point.		
Parameter's name	**Type**	**Description**
position	Vector3	User's touch event

Signature: void MouseUp(Vector position) **Description**: mouse position control		
Parameter's name	**Type**	**Description**

position	Vector3	User’s touch event

1.11 SoundTrackSelection:

Properties

Name	**Type**	**Visibility**	**Description**
uiCamera	Camera	public	User interface camera
buttonRenders	Renderer[]	public	Collection of button renders
buttonTexture	Texture[]	public	Collection of button Texture
hit	RaycastHit	public	Raycast hit
buyButton	GameObject	public	Buy button object
playButton	GameObject	public	Play button object
buyPopUpST	GameObject	public	
InAPPMenu	GameObject	public	In app menu iOS
InAPPMenuAndroid	GameObject	public	In app menu android
backButton	GameObject	public	Back button object
loadingLevelObj	GameObject	public	Loading level Object
menuObj	GameObject	public	Menu object
IAP	GameObject	public	
loadingscenes	GameObject	public	Loading scenes object
soundTrackIndex	int	public static	Soundtrack index
soundTracksObjs	GameObject[]	public	Soundtracks Object
soundTrackDisplayText	TextMesh	public	Display soundtrack text
soundTrackPriceDisplayText	TextMesh	public	Display soundtrack

			price text
headingText	TextMesh	public	Heading text

Operations

Signature: void Start () **Description**:		
Parameter's name	**Type**	**Description**
N/A		

Signature: void Update () **Description**: Update mouse position		
Parameter's name	**Type**	**Description**
N/A		

Signature: void MouseUp (Vector3 position) **Description**: mouse position control		
Parameter's name	**Type**	**Description**
position	Vector3	User's touch position

Signature: void ScreenPointToRay (Vector3 position) **Description**: Returns a ray going from camera through a screen point.		
Parameter's name	**Type**	**Description**
position	Vector3	User's touch position

Signature: void ShowPreviousSoundtrack () **Description**:		
Parameter's name	**Type**	**Description**
N/A		

Signature: void PurchaseSoundtrack () **Description**:		
Parameter's name	**Type**	**Description**
N/A		

Signature: void MouseDown (Vector3 position) **Description**: mouse position control		
Parameter's name	**Type**	**Description**
position	Vector3	User's touch position

Signature: void ShowSoundTrack () **Description**: Show soundtrack		
Parameter's name	**Type**	**Description**
N/A		

Signature: void ShowNextSoundtrack () **Description**: Show next soundtrack		
Parameter's name	**Type**	**Description**

N/A		

Signature: void ShowPreviousSoundtrack () **Description**: Show previous soundtrack		
Parameter's name	**Type**	**Description**
N/A		

Signature: void OnEnable () **Description**:		
Parameter's name	**Type**	**Description**
N/A		

Signature: void ShowSoundTrackINFO () **Description**: Show soundtrack information		
Parameter's name	**Type**	**Description**
N/A		

1.12 TotalCoins

Properties

Name	**Type**	**Visibility**	**Description**
totalCoins	int	public	Count total coins variable
coinsTxt	TextMesh	public	Display number of coins
staticInstance	TotalCoins	public static	

Operations

Signature: void Start () **Description**:		
Parameter's name	**Type**	**Description**
N/A		

Signature: void UpdateCoins () **Description**: Update and display total coins		
Parameter's name	**Type**	**Description**
N/A		

Signature: void AddCoins (int coins) **Description**: Add and update coins		
Parameter's name	**Type**	**Description**
coins	int	

Signature: void DeductCoins (int coins) **Description**: Deduct and update coins		
Parameter's name	**Type**	**Description**
coins	int	

Signature: void ClearCoins () **Description**: Clear and update number of coins		
Parameter's name	**Type**	**Description**
N/A		

Signature: void ClearAll () **Description**: Clear all and update number of highest coins		
Parameter's name	**Type**	**Description**
N/A		

1.13 TotalCoinsHighScore

Properties

Name	Type	Visibility	Description
totalCoinsHS	int	public static	Total high score coins count variable
maxCoinsHS	int	public static	Max coins high score count variable
coinsTxt	TextMesh	public	Display number of coins
maxcoinsTxt	TextMesh	public	Display number of max coins
staticInstance	TotalCoinsHighscore	public static	Static instance of this class

Operations

Signature: void Start () **Description**: Create static instance of this class		
Parameter's name	**Type**	**Description**
N/A		

Signature: void UpdateCoins () **Description**: Update coins and highest coins

Parameter’s name	Type	Description
N/A		

Signature: void ClearTotalCoinsHighscore (int coins) **Description**: Clear and update total and highest coins		
Parameter’s name	**Type**	**Description**
coins	int	

Signature: void ClearAll () **Description**: Clear all and update number of highest coins		
Parameter’s name	**Type**	**Description**
N/A		

1.14 TotalDistanceHighScore

Properties

Name	Type	Visibility	Description
totalDistanceHS	int	public static	Total high score distance count variable
maxDistanceHS	int	public static	Max high score coins count variable
distanceTxt	TextMesh	public	Display number of distance
maxdistanceTxt	TextMesh	public	Display number of max distance
staticInstance	TotalDistanceHighscore	public static	

Operations

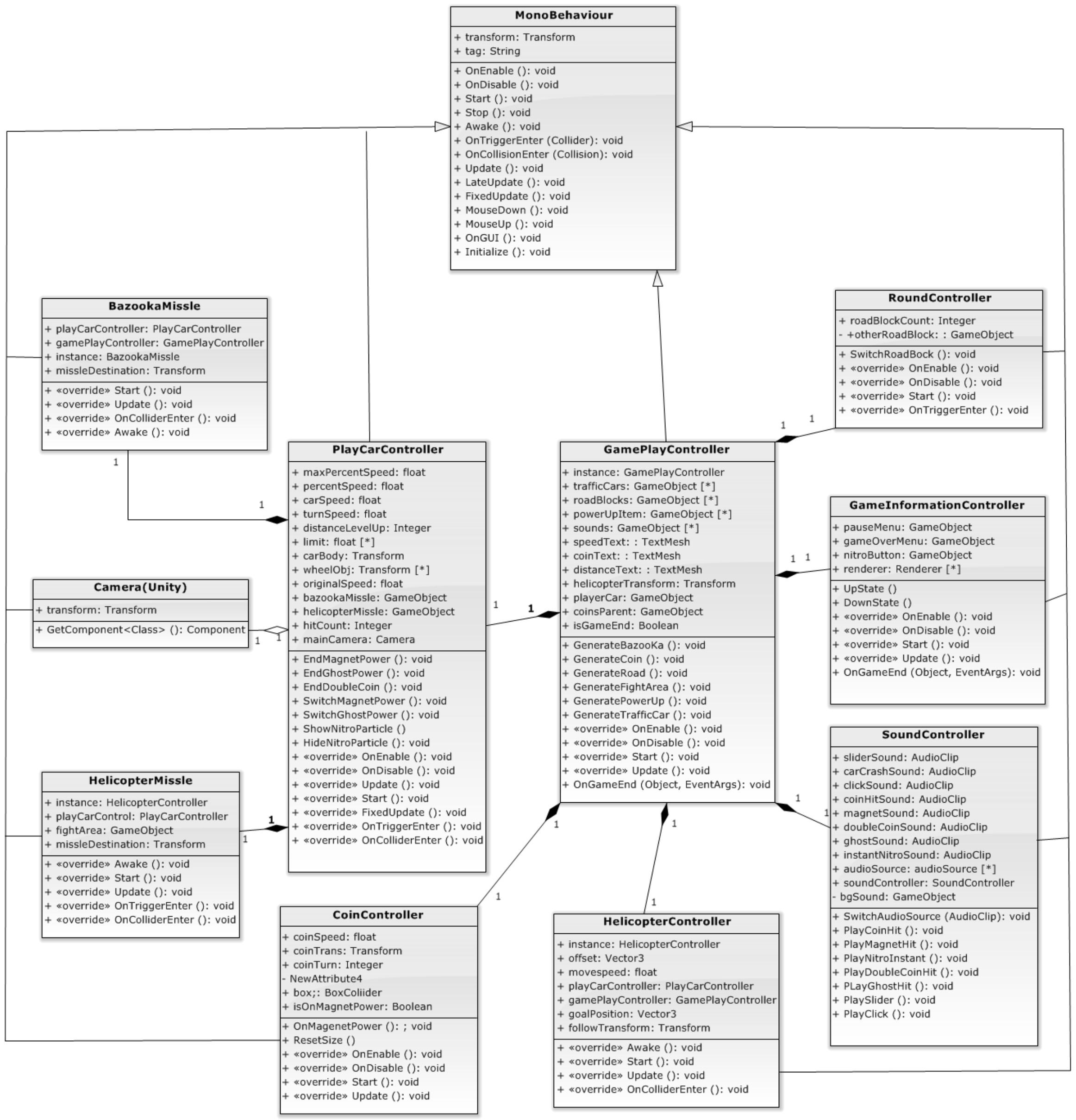
Signature: void Start ()		
Description: Create static instance of this class		
Parameter's name	**Type**	**Description**
N/A		

Signature: void UpdateDistance ()		
Description: Update and display highest distance		
Parameter's name	**Type**	**Description**
N/A		

Signature: void ClearTotalDistanceHighscore(int coins)		
Description: Clear and update highest distance		
Parameter's name	**Type**	**Description**
coins	int	

Signature: void ClearAll()		
Description: Delete all data		
Parameter's name	**Type**	**Description**
N/A		

GamePlayScene


MonoBehaviour
+ transform: Transform
+ tag: String
+ OnEnable (): void
+ OnDisable (): void
+ Start (): void
+ Stop (): void
+ Awake (): void
+ OnTriggerEnter (Collider): void
+ OnCollisionEnter (Collision): void
+ Update (): void
+ LateUpdate (): void
+ FixedUpdate (): void
+ MouseDown (): void
+ MouseUp (): void
+ OnGUI (): void
+ Initialize (): void
BazookaMissle
+ playCarController: PlayCarController
+ gamePlayController: GamePlayController
+ instance: BazookaMissle
+ missleDestination: Transform
+ «override» Start (): void
+ «override» Update (): void
+ «override» OnColliderEnter (): void
+ «override» Awake (): void
RoundController
+ roadBlockCount: Integer
- +otherRoadBlock: : GameObject
+ SwitchRoadBock (): void
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Start (): void
+ «override» OnTriggerEnter (): void
PlayCarController
+ maxPercentSpeed: float
+ percentSpeed: float
+ carSpeed: float
+ turnSpeed: float
+ distanceLevelUp: Integer
+ limit: float [*]
+ carBody: Transform
+ wheelObj: Transform [*]
+ originalSpeed: float
+ bazookaMissle: GameObject
+ helicopterMissle: GameObject
+ hitCount: Integer
+ mainCamera: Camera
+ EndMagnetPower (): void
+ EndGhostPower (): void
+ EndDoubleCoin (): void
+ SwitchMagnetPower (): void
+ SwitchGhostPower (): void
+ ShowNitroParticle ()
+ HideNitroParticle (): void
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Update (): void
+ «override» Start (): void
+ «override» FixedUpdate (): void
+ «override» OnTriggerEnter (): void
+ «override» OnColliderEnter (): void
GamePlayController
+ instance: GamePlayController
+ trafficCars: GameObject [*]
+ roadBlocks: GameObject [*]
+ powerUpItem: GameObject [*]
+ sounds: GameObject [*]
+ speedText: : TextMesh
+ coinText: : TextMesh
+ distanceText: : TextMesh
+ helicopterTransform: Transform
+ playerCar: GameObject
+ coinsParent: GameObject
+ isGameEnd: Boolean
+ GenerateBazooKa (): void
+ GenerateCoin (): void
+ GenerateRoad (): void
+ GenerateFightArea (): void
+ GeneratePowerUp (): void
+ GenerateTrafficCar (): void
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Start (): void
+ «override» Update (): void
+ OnGameEnd (Object, EventArgs): void
GameInformationController
+ pauseMenu: GameObject
+ gameOverMenu: GameObject
+ nitroButton: GameObject
+ renderer: Renderer [*]
+ UpState ()
+ DownState ()
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Start (): void
+ «override» Update (): void
+ OnGameEnd (Object, EventArgs): void
Camera(Unity)
+ transform: Transform
+ GetComponent<Class> (): Component
SoundController
+ sliderSound: AudioClip
+ carCrashSound: AudioClip
+ clickSound: AudioClip
+ coinHitSound: AudioClip
+ magnetSound: AudioClip
+ doubleCoinSound: AudioClip
+ ghostSound: AudioClip
+ instantNitroSound: AudioClip
+ audioSource: audioSource [*]
+ soundController: SoundController
- bgSound: GameObject
+ SwitchAudioSource (AudioClip): void
+ PlayCoinHit (): void
+ PlayMagnetHit (): void
+ PlayNitroInstant (): void
+ PlayDoubleCoinHit (): void
+ PLayGhostHit (): void
+ PlaySlider (): void
+ PlayClick (): void
HelicopterMissle
+ instance: HelicopterController
+ playCarControl: PlayCarController
+ fightArea: GameObject
+ missleDestination: Transform
+ «override» Awake (): void
+ «override» Start (): void
+ «override» Update (): void
+ «override» OnTriggerEnter (): void
+ «override» OnColliderEnter (): void
CoinController
+ coinSpeed: float
+ coinTrans: Transform
+ coinTurn: Integer
- NewAttribute4
+ box;: BoxColiider
+ isOnMagnetPower: Boolean
+ OnMagenetPower (): ; void
+ ResetSize ()
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Start (): void
+ «override» Update (): void
HelicopterController
+ instance: HelicopterController
+ offset: Vector3
+ movespeed: float
+ playCarController: PlayCarController
+ gamePlayController: GamePlayController
+ goalPosition: Vector3
+ followTransform: Transform
+ «override» Awake (): void
+ «override» Start (): void
+ «override» Update (): void
+ «override» OnColliderEnter (): void
1

1.15 CoinController

Propertites

Name	Type	Visibility	Description
coinSpeed	float	public	Coin’s speed

coinTrans	Transform	public	Coin’s transform
TurnCount	int	public static	Coin’s Y-axis degree
moveToPlayer	bool	public	Flag that detech coin and player impact event
thisTransform	Transform	public	Current coin’s transform
originalScale	Vector3	public	Original scale of coin
box	BoxCollider	public	Coins’component: BoxCollider
IsOnMagnetPower	bool	public static	Flag to check: Is Magnet effect activated.

Operation

Signature: void Start() **Description**: Give default value for coin’s properties		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void Update() **Description**: Set value of class’s properties after each frame.		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void OnEnable() **Description**: Findout coin’s component and register event		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void OnDisable() **Description**: Remove event.		
Parameter's name	**Type**	**Description**
N/A		

Signature: void OnGameEnd() **Description**: Destroy Object		
Parameter's name	**Type**	**Description**
N/A		

Signature: void OnMagnetPower() **Description**: Active magnet effect		
Parameter's name	**Type**	**Description**
obj	Object	
args	EventArgs	

Signature: void OffMagnetPower() **Description**: Stop magnet effect		
Parameter's name	**Type**	**Description**
N/A		

Signature: void ResetSize() **Description**: Set coin's size to original size		
Parameter's name	**Type**	**Description**

N/A		

1.16 GamePlayController

Properties

Name	**Type**	**Visibility**	**Description**
Instance	GamePlayController	public static	Instance of GamePlayControllerClass
CollectionsCoin Count	int	public static	Coin's collection
DistanceTravelled	int	public static	Distance that user have been geeting.
trafficCars	GameObject[]	public	Enemy cars in street
roadBlock	GameObject[]	public	To build road.
vlcCan	GameObject[]	public	To build rocks.
sideTree	GameObject[]	public	Current coin's transform
coinParent	GameObject[]	public	Previous appearing coin on street
powerPickUps	GameObject[]	public	PickUp Items
playerCars	GameObject[]	public	Player's cars
playerSoundTrack	GameObject[]	public	Sound is playing while game on.
missleArea	GameObject[]	public	Missle Area
playerObj	GameObject	public	Player's object
soundTrackObject	GameObject	public	Sound
gameEndMenu	GameObject	public	The menu that display when user want to stop playing game or lose game.

bazooka	GameObject	public	Special's item of car.
coinsText	TextMesh	public	Displays coins that player gain.
distanceText	TextMesh	public	Displays distance that player run.
gameOverText	TextMesh	public	Gameover message
mainCamera	Camera	public	Move to car carmera
droneChase	GameObject	public	Enemy drone
distanceDrone	int	private	Distance between Drone and player
distanceMissle	int	private	Distance between Missle and Drone
helicopterController	HelicopterController	public	Helicopter's Transform
isGameEnd	boolean	public	flag to detech game end

Operation

Signature: void Start()

Description: Set speed for car

Parameter's name	**Type**	**Description**
N/A		

Signature: void Update()

Description: updade value of class's properties after each frame.

Parameter's name	**Type**	**Description**
N/A		

Signature: void OnEnable() **Description**: Set value of class’s properties		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void OnDisable() **Description**: Repaint		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void GeneratePowerUpItem() **Description**: Create power up item in game		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void GenerateTrees **Description**: Create trees in street of game		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void GenerateRoad() **Description**: Create road of game		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void GenerateTrafficCar() **Description**: Create traffic car of game		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void GenerateBazooka() **Description**: Create bazooka weapon of game.		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void GenerateBazooka() **Description**: Create bazooka weapon of game.		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void GenerateFighArea() **Description**: Create area where the helicopter enemy attack		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void OnGameEnd(Object object, EventArgs event) **Description**: Destroy no longer unneeded game object		
Parameter’s name	**Type**	**Description**
object	Object	

event	EventArgs	

1.17 PlayerCarController

Properties

Name	**Type**	**Visibility**	**Description**
Instance	PlayerCarControl	public static	Instance of PlayerCarControl class
turnSpeed	float	public	Speed of car that follow Y-axis
carSpeed	float	public	Car’s speed
maxPercentSpeed	float	public	Max car’s speed
distanceLevelUp	int	public	Distance that car is running
tilt	float	public	Car’s tilt
limits	float[]	public	Car’s limits
gameEnded	EventHandler	public	Game end event handler
switchOnMagnetPower	EventHandler	public	Active magnet effect event handler
swtichOffMagnetPower	EventHandler	public	Stoop magnet effect event handler
magnetPowerTime	float	public	Magnet’s effect time
doublePowerTime	float	public	Double Coin effect time
ghostPowerTime	float	public	Ghost efffect time
isDoubleCoin	bool	public	Flag
nextFire	bool	public	Flag

isDoubleSpeed	bool	public	Flag
thisTrans	Transform	public	Car’s transform
particleParent	GameObject		Particle items that have child
thisPosition	Vector3	public	Car’s position
carMaterial	GameObject[]	public	To build car
originalCarSpeed	int	public	Default car speed
missleFighBack	HelicopterController	public	Helicopter’s missle
droneFight	HelicopterController	public	Drone enemy
hitCount	HelicopterController	public	The number of helicopter hitted
mainCamera	Camera	pubic	Third’s party and normal camera

Operations

Signature: void OnEnable()

Description: Initilize audio, item, effect, car.

Parameter’s name	**Type**	**Description**
N/A		

Signature: void Start()

Description: Generate enemy such as: traffic cars, helicopter and create road, power items

Parameter’s name	**Type**	**Description**
N/A		

Signature: void Update()

Description: Update class's properties after each frame		
Parameter's name	**Type**	**Description**
N/A		

Signature: void OnGameEnd() **Description**: Unregister event, set level, car to database		
Parameter's name	**Type**	**Description**
N/A		

Signature: void FixedUpdate() **Description**: Update's physics of game object.		
Parameter's name	**Type**	**Description**
N/A		

Signature: void OnTriggerEnter(Collider collider) **Description**: Detech event when the car player hit the power up item		
Parameter's name	**Type**	**Description**
collider	Collider	This is the component of game object that make collision when two collider overlap.

Signature: void OnCollisionEnter(Collision incomingCollision) **Description**: Handler when the car hit some object that have collider such as: traffic ar		
Parameter's name	**Type**	**Description**

incomingCollision	Collision	That happen when two collider impact each other

Signature: void ChangeShader1(Collision incomingCollision) **Description**: Create new effect for car		
Parameter's name	**Type**	**Description**
N/A		

Signature: void ChangeShader2(Collision incomingCollision) **Description**: Create new effect for car		
Parameter's name	**Type**	**Description**
N/A		

1.18 RoadController

Properties

Name	**Type**	**Visibility**	**Description**
justOnce	bool	public	A flag to check whether intance is generated or not.
RoadBlockCount	int	public static	To make next road block
otherRoadBlock	GameObject	public	Other road block

Operations

Signature: void OnEnable() **Description**: RegisterEvent for instance RoadGenarator		
Parameter's name	**Type**	**Description**

N/A		

Signature: void Start()

Description:

Parameter's name	Type	Description
N/A		

Signature: void OnTriggerEvent(Collider Collider)

Description: Detech car object in road and generate new road block

Parameter's name	Type	Description
N/A		

1.19 SoundController

Properties:

Name	Type	Visibility	Description
slider	AudioClip	public	Play when the car turns
clickSound	AudioClip	public	Play when user touch or click
carCrashSound	AudioClip	public	Play when the car crash
countHitSound	AudioClip	public	Play when the car hit coin.
magnetHitSound	AudioClip	public	Play when the car hit the magnet item.
doubleCoinHitSound	AudioClip	public	Play when the car hit

			the double coin item.
instantNitrousHitSound	AudioClip	public	Play when the car hit the nitrous item.
Static	SoundController	public static	Control the sound of game, instance of this class
audioSource	AudioSource[]	public	Sound source
bgSound	GameObject	public	Run while user playing game

Signature: void Start()

Description: Check level to decide background music will be activated or not

Parameter's name	**Type**	**Description**
N/A		

Signature: void SwitchAudioSource()

Description: Switchs audio.

Parameter's name	**Type**	**Description**
audio	AudioClip	Sound file

Signature: void PlayCrashSound()

Description: Play when the car crash

Parameter's name	**Type**	**Description**
N/A		

Signature: void PlayMagnetHit()

Description: Play when the car crash		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void PlayCoinHit () **Description**: Play when the car hit the coin item		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void PlayPowerPickUp() **Description**: Play when the car hit power pick up item		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void PlaySlider() **Description**: Play when the car slide on street		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void PlayDoubleHit() **Description**: Play when the car hit the double coin item		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void PlayNitrousHit() **Description**: Play when the car use nitro		
Parameter's name	**Type**	**Description**
N/A		

Signature: void PlayClick() **Description**: Play when user touch screen or click mouse		
Parameter's name	**Type**	**Description**
N/A		

1.20 GameInformationController

Properties:

Name	**Type**	**Visibility**	**Description**
pause	UIState(enum)	public	User's pause game state
resume	UIState(enum)	public	User's resume game state(game is playing)
gameOver	UIState(enum)	public	Game over state
empty	UIState(enum)	public	Initilizing state
ShieldTime	float	public static	Shield's time
MagnetTime	float	public static	Time of magnet effect
pauseMenu	GameObject	public	Displayed menu when user pause.
gameOverMenu	GameObject	public	Displayed menu when game over
coinImageContainer	GameObject	public	Sound source

distanceImageContainer	GameObject	public	Run while user playing game
pauseButton	GameObject	public	Pause button
nitrousUIParent	GameObject	public	Using nitro button
loadingAdmob	GameObject	public	Load admob
raycastHit	RaycastHit	public	Raycast hit
buttonTex	Texture[]	public	Button texture
pauseButtonTex	Texture[]	public	Pause button texture
brakeButtonTex	Texture[]	public	Brake button texture
nitroButtonTex	Texture[]	public	Nitro button texture
pauseButtonRenderer	Renderer	public	Pause button texture renderer
nitrousButtonRenderer	Renderer	public	Nitro button texture reanderer
buttonRenderers	Renderer[]	public	Collection of button renderer
nitrousTransform	Transform	public	Nitro' transform

Operation

Signature: void OnEnable()

Description: Register event

Parameter's name	**Type**	**Description**
N/A		

Signature: void OnDisable()

Description: Unregister event

Parameter’s name	**Type**	**Description**
N/A		

Signature: void Start() **Description**: Detech device, setup class’s properties		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void Update() **Description**: Listen user’s event(touch, click), update value, image after each frame		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void OnEnable() **Description**: Register event		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void OnGameEnd(Object object, EventArgs args) **Description**: Stop renderer button, stop racing scence.		
Parameter’s name	**Type**	**Description**
Obj	Object	
arsg	EventArgs	

Signature: void DownState(Vector3 position) **Description**: Handler when user's control		
Parameter's name	**Type**	**Description**
postion	Vector3	Catch the postion that user touch or click.
Signature: void UpState(Vector3 position) **Description**: Handler when user's control		
Parameter's name	**Type**	**Description**
postion	Vector3	Catch the postion that user touch or click.

1.21 HelicopterMissle

Properties

Name	**Type**	**Visibility**	**Description**
Instance	HelicopterMissle	public static	Instance of this class
playercarcontrol	PlayerCarControl	private	Control car's player
FightArea	GameObject	private	Area of Fight
missileDestination	Transform	private	Destination of Missile
expos	GameObject	public	

Operations

Signature: void Awake() **Description**: Call Dronefight instance		
Parameter's name	**Type**	**Description**
N/A		

Signature: void Start() **Description**: Give default value for Drone’s properties		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void Update() **Description**: Update value of class’s properties after each frame		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void OnTriggerEnter (Collider collider) **Description**: Fight area and Missile destination trigger		
Parameter’s name	**Type**	**Description**
collider	Collider	

Signature: void Destroy () **Description**: Removes a gameObject		
Parameter’s name	**Type**	**Description**
gameObject	GameObject	

Signature: void OnCollisionEnter(Collision incomingCollision) **Description**: On-collison event, destroy traffic car or player’s cars		
Parameter’s name	**Type**	**Description**

incomingCollision	Collision	

1.22 BazookaMissle

Properties

Name	**Type**	**Visibility**	**Description**
Instance	BazookaMissle	public static	Instance of this class
playercarcontrol	PlayerCarControl	private	Player car control
gameplaycontrol	GamePlayController	private	Game play control
missileDestination	Transform	private	Destination of Missile
expos	GameObject	public	

Operations

Signature: void Awake()

Description: Call Drone fight back instance

Parameter's name	**Type**	**Description**
N/A		

Signature: void Start()

Description: Give default value for Drone and car's properties

Parameter's name	**Type**	**Description**
N/A		

Signature: void OnCollisionEnter(Collision incomingCollision)

Description: On-collision event, player's car fight back to drone

Parameter's name	**Type**	**Description**

collision	Collision	

Signature: void Destroy () **Description**: Removes a gameObject		
Parameter's name	**Type**	**Description**
gameObject	GameObject	

Signature: void CancelFire () **Description**: Cancel fire		
Parameter's name	**Type**	**Description**
N/A		

1.23 HelicopterController:

Properties

Name	**Type**	**Visibility**	**Description**
Instance	HelicopterController	public static	Instance of HelicopterController Class
followTransform	Transform	private	Follow player'car transform
offset	Vector3	public	Current coordinate
moveSpeed	float	public	Number of move speed variable
rotationspeed	float	public	Rotation speed variable
playercarcontrol	PlayerCarControl	private	Player car control
gameplaycontrol	GamePlayController	private	Game play control
goalPos	Vector3	public	Goal Position

distance	int	public	The distance of helicopter and player's car
xAxisDis	int	public	

Operations

Signature: void Awake() **Description**: Call HelicopterController instance		
Parameter's name	**Type**	**Description**
N/A		

Signature: void Start() **Description**: Give default value for Helicopter and player control's properties		
Parameter's name	**Type**	**Description**
N/A		

Signature: void OnCollisionEnter(Collision incomingCollision) **Description**: On-collision helicopter		
Parameter's name	**Type**	**Description**
incomingCollision	Collision	

Signature: void CancelFire() **Description**: Cancel fire		
Parameter's name	**Type**	**Description**
N/A		

Signature: void Destroy() **Description**: Removes a gameObject		
Parameter's name	**Type**	**Description**
gameObject	GameObject	

Signature: void Update () **Description**: Update position of player's car and helicopter each frame		
Parameter's name	**Type**	**Description**
N/A		

2. MultiPlayer

2.1 ConnectionScene

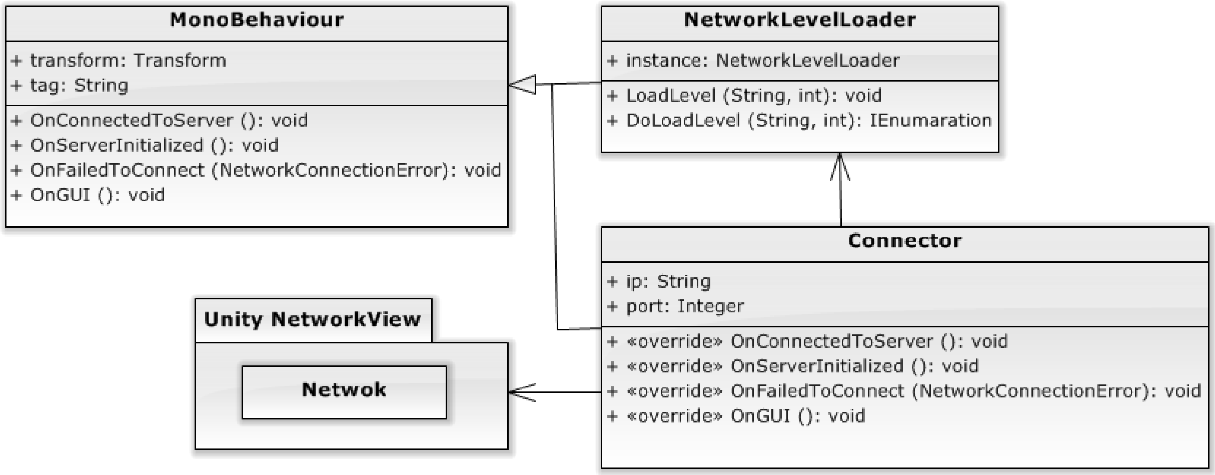

MonoBehaviour
+ transform: Transform
+ tag: String
+ OnConnectedToServer (): void
+ OnServerInitialized (): void
+ OnFailedToConnect (NetworkConnectionError): void
+ OnGUI (): void
NetworkLevelLoader
+ instance: NetworkLevelLoader
+ LoadLevel (String, int): void
+ DoLoadLevel (String, int): IEnumaration
Connector
+ ip: String
+ port: Integer
+ «override» OnConnectedToServer (): void
+ «override» OnServerInitialized (): void
+ «override» OnFailedToConnect (NetworkConnectionError): void
+ «override» OnGUI (): void
Unity NetworkView
Netwok

Figure 4.21: ConnectionScene

2.1.1 Connector

Properties

Name	**Type**	**Visibility**	**Description**

ip	String	public	Ip of server that make a host
port	int	public	Port of server
isConnected	boolean	public	Flag to check whether server and client have already connected

Operations:

Signature: void OnGUI()

Description: Make the connection scene and connect to server

Parameter's name	**Type**	**Description**
N/A		

Signature: void OnConnectedToServer()

Description: Create NetworkLevelLoader

Parameter's name	**Type**	**Description**
N/A		

Signature: void OnServerInitialized()

Description: Create scene game

Parameter's name	**Type**	**Description**
N/A		

Signature: OnFailedToConnect(NetworkConnectionError error)

Description: Create NetworkLevelLoader

Parameter's name	**Type**	**Description**

error	NetworkConnectionError	Description of network error

2.1.2 NetworkLevelLoader

Properties

Name	**Type**	**Visibility**	**Description**
instance	NetworkLevelLoader	public	Ip of server that make a host

Operations

Signature: void LoadLevel(String levelName, int prefix)

Description: Start coroutine.

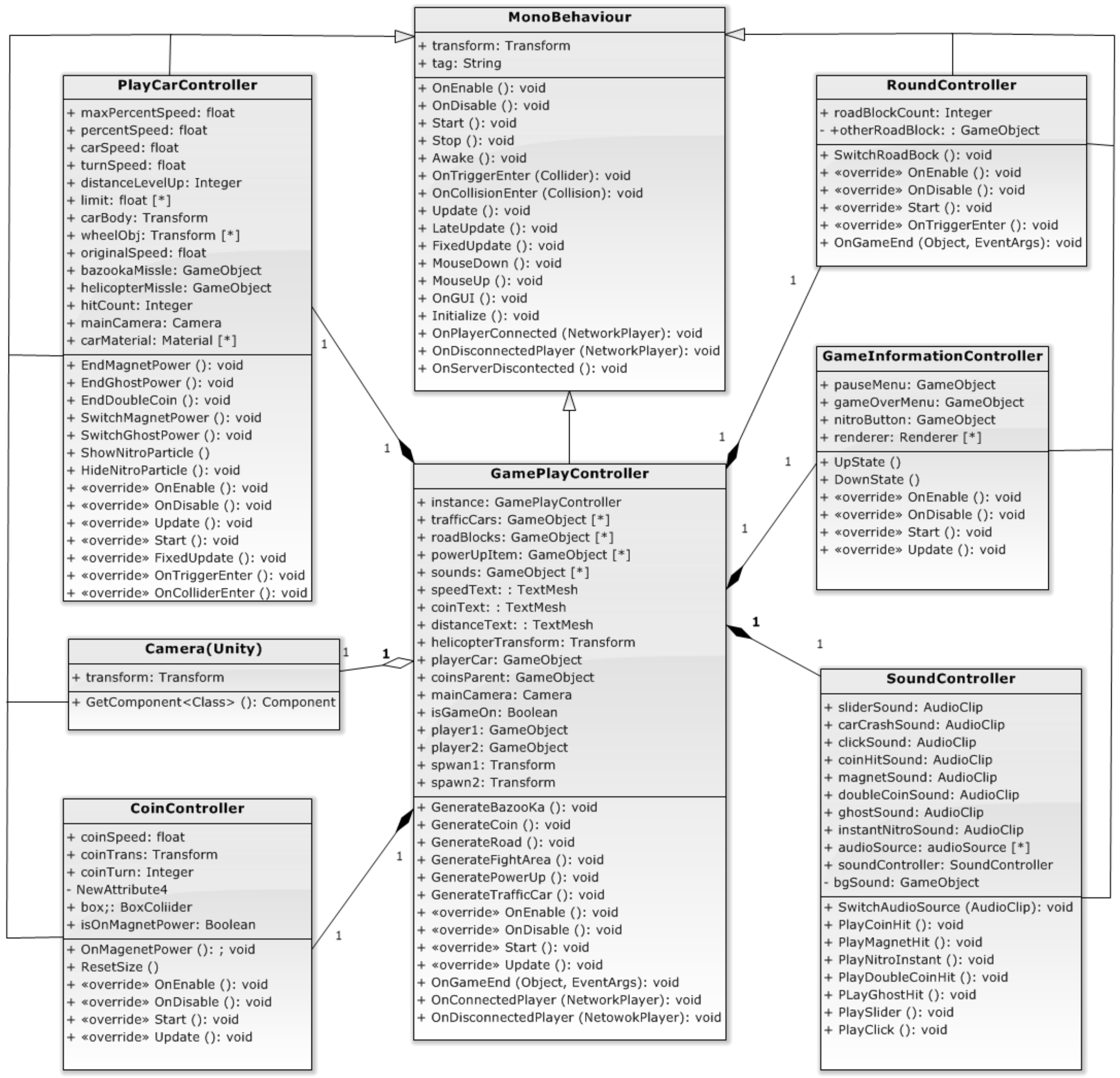
Parameter's name	**Type**	**Description**
levelName	String	Level's name
int	prefix	

Signature: IEnumerator DoLoadLevel(String levelName, int prefix)

Description: Setup network's properties to prepare connect

Parameter's name	**Type**	**Description**
levelName	String	Level's name
int	prefix	

2.2 GamePlayScene


MonoBehaviour
+ transform: Transform
+ tag: String
+ OnEnable (): void
+ OnDisable (): void
+ Start (): void
+ Stop (): void
+ Awake (): void
+ OnTriggerEnter (Collider): void
+ OnCollisionEnter (Collision): void
+ Update (): void
+ LateUpdate (): void
+ FixedUpdate (): void
+ MouseDown (): void
+ MouseUp (): void
+ OnGUI (): void
+ Initialize (): void
+ OnPlayerConnected (NetworkPlayer): void
+ OnDisconnectedPlayer (NetworkPlayer): void
+ OnServerDiscontected (): void
PlayCarController
+ maxPercentSpeed: float
+ percentSpeed: float
+ carSpeed: float
+ turnSpeed: float
+ distanceLevelUp: Integer
+ limit: float [*]
+ carBody: Transform
+ wheelObj: Transform [*]
+ originalSpeed: float
+ bazookaMissle: GameObject
+ helicopterMissle: GameObject
+ hitCount: Integer
+ mainCamera: Camera
+ carMaterial: Material [*]
+ EndMagnetPower (): void
+ EndGhostPower (): void
+ EndDoubleCoin (): void
+ SwitchMagnetPower (): void
+ SwitchGhostPower (): void
+ ShowNitroParticle ()
+ HideNitroParticle (): void
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Update (): void
+ «override» Start (): void
+ «override» FixedUpdate (): void
+ «override» OnTriggerEnter (): void
+ «override» OnColliderEnter (): void
RoundController
+ roadBlockCount: Integer
- +otherRoadBlock: : GameObject
+ SwitchRoadBock (): void
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Start (): void
+ «override» OnTriggerEnter (): void
+ OnGameEnd (Object, EventArgs): void
GameInformationController
+ pauseMenu: GameObject
+ gameOverMenu: GameObject
+ nitroButton: GameObject
+ renderer: Renderer [*]
+ UpState ()
+ DownState ()
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Start (): void
+ «override» Update (): void
GamePlayController
+ instance: GamePlayController
+ trafficCars: GameObject [*]
+ roadBlocks: GameObject [*]
+ powerUpItem: GameObject [*]
+ sounds: GameObject [*]
+ speedText: : TextMesh
+ coinText: : TextMesh
+ distanceText: : TextMesh
+ helicopterTransform: Transform
+ playerCar: GameObject
+ coinsParent: GameObject
+ mainCamera: Camera
+ isGameOn: Boolean
+ player1: GameObject
+ player2: GameObject
+ spwan1: Transform
+ spawn2: Transform
+ GenerateBazooKa (): void
+ GenerateCoin (): void
+ GenerateRoad (): void
+ GenerateFightArea (): void
+ GeneratePowerUp (): void
+ GenerateTrafficCar (): void
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Start (): void
+ «override» Update (): void
+ OnGameEnd (Object, EventArgs): void
+ OnConnectedPlayer (NetworkPlayer): void
+ OnDisconnectedPlayer (NetowokPlayer): void
Camera(Unity)
+ transform: Transform
+ GetComponent<Class> (): Component
CoinController
+ coinSpeed: float
+ coinTrans: Transform
+ coinTurn: Integer
- NewAttribute4
+ box;: BoxColiider
+ isOnMagnetPower: Boolean
+ OnMagenetPower (): ; void
+ ResetSize ()
+ «override» OnEnable (): void
+ «override» OnDisable (): void
+ «override» Start (): void
+ «override» Update (): void
SoundController
+ sliderSound: AudioClip
+ carCrashSound: AudioClip
+ clickSound: AudioClip
+ coinHitSound: AudioClip
+ magnetSound: AudioClip
+ doubleCoinSound: AudioClip
+ ghostSound: AudioClip
+ instantNitroSound: AudioClip
+ audioSource: audioSource [*]
+ soundController: SoundController
- bgSound: GameObject
+ SwitchAudioSource (AudioClip): void
+ PlayCoinHit (): void
+ PlayMagnetHit (): void
+ PlayNitroInstant (): void
+ PlayDoubleCoinHit (): void
+ PLayGhostHit (): void
+ PlaySlider (): void
+ PlayClick (): void
1
1
1
1
1
1
1
1
1
1
1
1
1
1

We reuse three class: CoinController, RoadController, SoundController (III.Detail Description of Compomnent, SinglePlayer, GamePlayScene)

2.2.1 GamePlayControllerMultiplayer

Properties

Name	Type	Visibility	Description
Instance	GamePlayController	public static	Instance of GamePlayControllerClass
CollectionsCoin Count	int	public static	Coin’s collection
DistanceTravelled	int	public	Distance that user have

		static	been geeting.
trafficCars	GameObject[]	public	Enemy cars in street
roadBlock	GameObject[]	public	To build road.
vlcCan	GameObject[]	public	To build rocks.
sideTree	GameObject[]	public	Current coin’s transform
coinParent	GameObject[]	public	Previous appearing coin on street
powerPickUps	GameObject[]	public	PickUp Items
playerCars	GameObject[]	public	Player’s cars
playerSoundTrack	GameObject[]	public	Sound is playing while game on.
missleArea	GameObject[]	public	Missle Area
playerObj	GameObject	public	Player’s object
soundTrackObject	GameObject	public	Sound
gameEndMenu	GameObject	public	The menu that display when user want to stop playing game or lose game.
bazooka	GameObject	public	Special’s item of car.
coinsText	TextMesh	public	Displays coins that player gain.
distanceText	TextMesh	public	Displays distance that player run.
gameOverText	TextMesh	public	Gameover message
mainCamera	Camera	public	Move to car carmera
droneChase	GameObject	public	Enemy drone
distanceDrone	int	private	Distance between Drone and player
distanceMissle	int	private	Distance between Missle

			and Drone
helicopterController	HelicopterController	public	Helicopter’s Transform
player1	GameObject	public	Instance of player1 game object
player2	GameObject	public	Instance of player2 game object
spawn1	Transform	public	Spwan position
spawn2	Transform	public	Spwan position

Operation

Signature: void Start() **Description**: Set speed for car		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void OnGameEnd(Object object, EventArsg event) **Description**: Destroy no longer unneeded game object		
Parameter’s name	**Type**	**Description**
object	Object	
event	EventArgs	

Signature: void Update() **Description**: updade value of class’s properties after each frame.		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void OnEnable() **Description**: Set value of class's properties		
Parameter's name	**Type**	**Description**
N/A		

Signature: void OnDisable() **Description**: Repaint		
Parameter's name	**Type**	**Description**
N/A		

Signature: void GeneratePowerUpItem() **Description**: Create power up item in game		
Parameter's name	**Type**	**Description**
N/A		

Signature: void GenerateTrees **Description**: Create trees in street of game		
Parameter's name	**Type**	**Description**
N/A		

Signature: void GenerateRoad() **Description**: Create road of game		
Parameter's name	**Type**	**Description**
N/A		

Signature: void GenerateTrafficCar() **Description**: Create traffic car of game		
Parameter's name	**Type**	**Description**
N/A		

Signature: void GenerateBazooka() **Description**: Create bazooka weapon of game.		
Parameter's name	**Type**	**Description**
N/A		

Signature: void GenerateBazooka() **Description**: Create bazooka weapon of game.		
Parameter's name	**Type**	**Description**
N/A		

Signature: void GenerateFighArea() **Description**: Create area where the helicopter enemy attack		
Parameter's name	**Type**	**Description**
N/A		

Signature: void OnConnectedPlayer(NetworkPlayer player) **Description**: Call function to connected player by RPC		
Parameter's name	**Type**	**Description**
player	NetworkPlayer	

Signature: void OnDisconnectedPlayer(NetworkPlayer player) **Description**: Destroy connected player object		
Parameter's name	**Type**	**Description**
player	NetworkPlayer	

Signature: void OnDisconnectedFromServer() **Description**: Return to main menu		
Parameter's name	**Type**	**Description**
N/A		

2.2.2 PlayerCarControllerMultiplayer

Properties

Name	**Type**	**Visibility**	**Description**
Instance	PlayerCarControl	public static	Instance of PlayerCarControl class
turnSpeed	float	public	Speed of car that follow Y-axis
carSpeed	float	public	Car's speed
maxPercentSpeed	float	public	Max car's speed
distanceLevelUp	int	public	Distance that car is running
tilt	float	public	Car's tilt
limits	float[]	public	Car's limits
gameEnded	EventHandler	public	Game end event handler

switchOnMagnetPower	EventHandler	public	Active magnet effect event handler
swtichOffMagnetPower	EventHandler	public	Stoop magnet effect event handler
magnetPowerTime	float	public	Magnet's effect time
doublePowerTime	float	public	Double Coin effect time
ghostPowerTime	float	public	Ghost efffect time
isDoubleCoin	bool	public	Flag
nextFire	bool	public	Flag
isDoubleSpeed	bool	public	Flag
thisTrans	Transform	public	Car's transform
particleParent	GameObject		Particle items that have child
thisPosition	Vector3	public	Car's position
carMaterial	GameObject[]	public	To build car
originalCarSpeed	int	public	Default car speed
missleFighBack	HelicopterController	public	Helicopter's missle
droneFight	HelicopterController	public	Drone enemy
hitCount	HelicopterController	public	The number of helicopter hitted
mainCamera	Camera	pubic	Third's party and normal camera

Operations

Signature: void OnEnable()

Description: Initilize audio, item, effect, car.

Parameter's name	**Type**	**Description**

N/A		

Signature: void Start() **Description**: Generate enemy such as: traffic cars, helicopter and create road, power items		
Parameter's name	**Type**	**Description**
N/A		

Signature: void Update() **Description**: Update class's properties after each frame		
Parameter's name	**Type**	**Description**
N/A		

Signature: void OnGameEnd() **Description**: Unregister event, set level, car to database		
Parameter's name	**Type**	**Description**
N/A		

Signature: void FixedUpdate() **Description**: Update's physics of game object.		
Parameter's name	**Type**	**Description**
N/A		

Signature: void OnTriggerEnter(Collider collider) **Description**: Detech event when the car player hit the power up item

Parameter's name	Type	Description
collider	Collider	This is the component of game object that make collision when two collider overlap.

Signature: void OnCollisionEnter(Collision incomingCollision) **Description**: Handler when the car hit some object that have collider such as: traffic ar		
Parameter's name	**Type**	**Description**
incomingCollision	Collision	That happen when two collider impact each other

Signature: void ChangeShader2(Collision incomingCollision) **Description**: Create new effect for car		
Parameter's name	**Type**	**Description**
N/A		

2.2.3 GameInformationControllerMultiplayer

Properties:

Name	Type	Visibility	Description
pause	UIState(enum)	public	User's pause game state
resume	UIState(enum)	public	User's resume game state(game is playing)
gameOver	UIState(enum)	public	Game over state
empty	UIState(enum)	public	Initilizing state

ShieldTime	float	public static	Shield’s time
MagnetTime	float	public static	Time of magnet effect
pauseMenu	GameObject	public	Displayed menu when user pause.
gameOverMenu	GameObject	public	Displayed menu when game over
coinImageContainer	GameObject	public	Sound source
distanceImageContainer	GameObject	public	Run while user playing game
pauseButton	GameObject	public	Pause button
nitrousUIParent	GameObject	public	Using nitro button
loadingAdmob	GameObject	public	Load admob
raycastHit	RaycastHit	public	Raycast hit
buttonText	Texture[]	public	Button texture
pauseButtonText	Texture[]	public	Pause button texture
nitroButtonText	Texture[]	public	Nitro button texture
pauseButtonRenderer	Renderer	public	Pause button texture renderer
nitrousButtonRenderer	Renderer	public	Nitro button texture reanderer
buttonRenderers	Renderer[]	public	Collection of button renderer
nitrousTransform	Transform	public	Nitro’ transform

Operation

Signature: void OnEnable()

Description: Register event

Parameter's name	Type	Description
N/A		

Signature: void OnDisable()

Description: Unregister event

Parameter's name	Type	Description
N/A		

Signature: void Start()

Description: Detech device, setup class's properties

Parameter's name	Type	Description
N/A		

Signature: void Update()

Description: Listen user's event(touch, click), update value, image after each frame

Parameter's name	Type	Description
N/A		

Signature: void OnEnable()

Description: Register event

Parameter's name	Type	Description
N/A		

Signature: void OnGameEnd(Object object, EventArgs args)

Description: Stop renderer button, stop racing scence.

Parameter's name	Type	Description
Obj	Object	
arsg	EventArgs	

Signature: void DownState(Vector3 position) **Description**: Handler when user's control		
Parameter's name	**Type**	**Description**
postion	Vector3	Catch the postion that user touch or click.

Signature: void UpState(Vector3 position) **Description**: Handler when user's control		
Parameter's name	**Type**	**Description**
postion	Vector3	Catch the postion that user touch or click.

3. Utilities

3.1 Game Data

Properties

Name	Type	Visibility	Description
CurrentLevel	int	public static	Current game's level
PercenLevel	float	public static	When percent become 100, user will go to next level
ExperientOnRun	float	public	Distance that user gets.

		static	
ExperientOnLevelBegin	float	public static	Default experience.

Operations

Signature: void Save() **Description**: Save data to player’s preference		
Parameter’s name	**Type**	**Description**
N/A		

3.2 Destroyer

Properties
N/A

Operations

Signature: void OnBecameInvisible() **Description**: Destroy object when it’s invisible(No longer exsited in screen)		
Parameter’s name	**Type**	**Description**
N/A		

Signature: void Destroy(GameObject obj) **Description**: Destroy object when it’s invisible(No longer exsited in screen)		
Parameter’s name	**Type**	**Description**
obj	GameObject	

3.3 GameCenterSingleton

Properties

Name	**Type**	**Visibility**	**Description**
instance	GameCenterSingleton	public static	Instance of GameCenterSingleton class
achievements	IAchievement[]	private	Achievement collection

Operations

Signature: void Initialize() **Description**: Initialize user information		
Parameter's name	**Type**	**Description**
N/A		

Signature: bool IsUserAuthenticated () **Description**: check user authentication		
Parameter's name	**Type**	**Description**
N/A		

Signature: void ShowAchievementUI () **Description**: show user's achievement		
Parameter's name	**Type**	**Description**
N/A		

Signature: void ShowLeaderboardUI () **Description**: show leader board		
Parameter's name	**Type**	**Description**
N/A		

Signature: void ReportScore (long score, String leaderBoardUI) **Description**: Report player's score on leaderboard		
Parameter's name	**Type**	**Description**
score	long	Score
leaderBoardID	string	ID of leander board

Signature: bool AddAchievementProgress (String achivementID, foat percentageToAdd) **Description**: check achievement progress adding		
Parameter's name	**Type**	**Description**
achievementID	string	ID of Achivement
percentageToAdd	float	Percentage of achivement

Signature: bool ReportAchievementProgress (String achivementID, foat progressCompleted) **Description**: check achievement progress reporting		
Parameter's name	**Type**	**Description**
achievementID	string	ID of Achivement
progressCompleted	float	

Signature: void ResetAchievements () **Description**: Reset user achievement		
Parameter's name	**Type**	**Description**
N/A		

Signature: void LoadAchievements ()

Description: Load user achievement

Parameter’s name	**Type**	**Description**
N/A		

Signature: void ProcessAuthentication (Boolean success)

Description: Authentication Process

Parameter’s name	**Type**	**Description**
success	Boolean	

Signature: void ProcessLoadedAchievements ()

Description: Update achievement process

Parameter’s name	**Type**	**Description**
achievements	IAchievement[]	

Signature: bool IsAchievementComplete (String achivementID)

Description: check achievement completion

Parameter’s name	**Type**	**Description**
achievementID	string	ID of achivement

Signature: IAchievement GetAchievement (String achivementID)

Description: Get achievement

Parameter’s name	**Type**	**Description**
achievementID	string	

Signature: void ResetAchievementsHandler (Boolean status) **Description**: Reset user’s achievement		
Parameter’s name	**Type**	**Description**
status	bool	

Signature: void ReportAchievementProgress (String achivementID, float percentageToAdd) **Description** Reports the progress of an achievement.		
Parameter’s name	**Type**	**Description**
achievementID	string	ID of Achivement
percentageToAdd	float	

Chapter 5 – テスト計画

この部分では、主な機能のテストケースが含まれている。他のテストケースは、他の文書がある。

I. はじめに

1. 項目

このドキュメントでは、システムの主な機能のテスト計画を説明します。ここでテスト手法を中心に説明する。

２番目のセクションでは、テストの要求を説明する。これはテスト対象の機能およびテストの種類を使用する必要がある。

３番目のセクションでは、単体テスト、積分テスト、UI テストなどのテスト種類を証明する。

４番目では、テストケースを作ることです。

５番目はテストのマイルストーンの概要を記載する。

６番目では、テスト文書の配達を述べる。

2. 定義、頭字語および略語

定義/頭字語/略語	説明	備考
AT	受入テスト	
DMS	不具合管理システム(Fsoft ツール)	
IT	結合テスト	
PM	プロジェクトマネージャ	
PTL	プロジェクト技術リーダ	
QA	品質保証部	
SRS	ソフトウェア要求仕様書	
ST	システムテスト	

定義/頭字語/略語	説明	備考
TP	テスト計画書	
TC	テストケース	
TR	テスト報告書	
UAT	ユーザ受入テスト	
UT	単体テスト	

3. 参照資料

ドキュメント名	作成者	版数	発行日
N/A			

4. 背景情報

テスト対象は、クラス、オブジェクトなどに関して、実行します。それぞれの機能は、独自の目標がある。たとえば、アイテムを拾って、車の移動可能を高めて、敵を絶滅することができます。

5. テスト範囲

まず、各の開発者は、モジュールの単体テストを進む。

それで、すべてのチームメンバーの支援を受けて、チームリーダは統合テストを実行します。

次に、システムが要件を満たしていることを確認するために、チームはシステムテストを進む。

最後に、監督者は受入テストを進む。

6. 制約事項

N/A

7. リスク一覧

N/A

8. 必要となるトレーニング

単体テストの使用される方法：ブラックボックステスト。

II. テスト要求

1. テスト項目

これからブラックボックステスト方法で実施して、必要な機能のリストです。

- マップを表す
- マップを選ぶ
- マップをアンロック
- 車を表す
- 車を選ぶ
- 車をアンロック
- 車を調節する
- スピードブースト
- アイテムを拾う
  ＋スルーに運動
  ＋コインを二倍に受ける
  ＋磁石
  ＋バズーカ
  ＋スーパー
  ＋瞬時エネルギー
- 奏功シェア
- 高得点を表す

2. テスト受入基準

テストの許容範囲は：

- コードカバレッジ: 95%
- テストケースの数/ コードの行 (K): 100
- 分岐カバレッジ: 100%
- パスカバレッジ: 100%

3. テスト戦略

単体テストの一般的な戦略は、1/2/4 率である。 1 通常、 2 境界と 4 異常なケースである。

完了基準：

- すべてケースは、正常にテストされていた。
- すべて欠陥が対処して、修正された。

特別な条件：

- 時間がなくなるとき、テストが停止される。
- テストは、特定のカバレッジに達する。

4. テストタイプ

4.1 機能テスト

テスト目的	各機能のデータ入力、データ処理、アクション出力を確認する。
方法	仮定アクションで、テストケースを実行して、このリストを確認する。 • 有効アクションの時、期待された結果が発生する • 無効アクションの時、期待されたエラーが発生する
完了基準	すべてのケースが実行された。 すべての欠陥が修正された。
特記事項	

4.2 ユーザインタフェーステスト

テスト目的	• テスト対象のナビゲーションが適切にビジネス機能と要件を表示している。 • メニューや、サイズや状態などのオブジェクトが標準に準拠する。
方法	オブジェクトのナビゲーションを確認するため、テストケースを作り、変更する。
完了基準	各のオブジェクトが標準に準拠する。
特記事項	

4.3 性能テスト

テスト目的	パフォーマンスの行動下記の条件を確認する： • 通常な作業負荷 • 最悪の場合の作業負荷
方法	• テストの手順を使用する。 • 取引の数を高めるため、データファイルを変更する。
完了基準	テストスクリプトがうまく実行される。
特記事項	

4.4 回帰テスト

テスト目的	回帰テストでは、変更がエラーを発生しないのを確認するため、変更したソフトウェアの部分を検査する。
方法	• 既存なテストスイートから、テストケースのセットを再利用する。 • 合理的な UTT を使用する。 （　既存なテストケースから、無作為に８０％テ

	ストケースを選択する)
完了基準	すべてのケースを実行した。
特記事項	

4.5 テストステージ

テストタイプ	テストステージ			
	単体	結合	総合	受入
<機能テスト>	X	X	X	X
<ユーザインターフェース テスト>	X		X	
<アクション整合性テスト>		X	X	

4.6 ツール

目的	ツール	ベンダー/社内	版数
欠陥のログ	<DMS>		<3.8.1>
ユニットテスト	<UTT>		<4.6>

5. リソース

5.1 人的リソース

下表にテストメンバーの担当作業を示す。

担当者名	作業内容/備考
TrieuTTH	テスト計画を作る。 単体テストケースを作る。 テストケースをレビューする。 テスト報告書を作る。

CuongNC	単体テストケースを作る。 テストケースを実行する。 テスト結果を伝える。
DuyTQ	単体テストケースを作る。 テストケースを実行する。 テスト結果を伝える。

5.2 システム

ソフトウェアツル：UTT v4.6, DMS v8.3.1

III. 実行

ブラックボックステストを中心する方法で機能テストを実行する。

＊機能テスト

1. マップを表す

主流：

ー車を選ぶ後ユーザがマップを選ばなければなりません。

ID	説明	期待結果	実際結果	ケース結果
1	前提条件：ユーザが車を選ぶ画面で次ボタンを押す。	ーマップを表示することができる。	ーマップを表示することができる。	成功
2	ーユーザがマップを選ぶ	ー次マップを表示することができる。	ー次マップがありませんでした。（一つだけ）	失敗

2. マップを選ぶ

主流：

ー車を選ぶ後ユーザがマップを選ばなければなりません。

ID	説明	期待結果	実際結果	ケース結果
3	前提条件：ユーザが車を選ぶ画面で次ボタンを押して、ほしいマップをクリックします。	ー正確にユーザの選んだマップを表することができる。	ー正確にユーザの選んだマップを表することができる。	成功
4	前提条件：ユーザが車を選ぶ画面で次ボタンを押して、マップを探して、ほしいマップをクリックします。	ー正確にユーザの選んだマップを表することができる。	ー正確にユーザの選んだマップを表することができる。	成功
5	前提条件：ユーザはマップを選んだ。	ーゲームプレイで正しいマップを表示する。	ーゲームプレイで正しいマップを表示する。	成功

3. マップをアンロック

主流：

ーユーザがマップを見て、コインを使用してマップをアンロックすることができます

ID	説明	期待結果	実際結果	ケース結果
6	前提条件：ユーザはマップを表す画面がいる。 ーコインが足りる場合はアンロックボタンを押す	ー新しいマップをアンロックすることができる。	ー新しいマップをアンロックすることができる。	成功

7	前提条件：ユーザはマップを表す画面がいる。 ーコインが足りない場合はアンロックボタンを押す	ー新しいマップをアンロックすることができない。	ー新しいマップをアンロックすることができない。	成功

4. 車を表す

主流：

ーユーザがマインメニュの画面でプレイボタンを押す。

ーシステムが各車を並んで表示します。

ID	**説明**	**期待結果**	**実際結果**	**ケース結果**
8	前提条件：ユーザがマインメニュ画面でプレイボタンを押す。	ー車を表示することができる。	ー車を表示することができる。	成功
9	ーユーザが車を選ぶ	ー次車を表示することができる。	ー次車が表示することができる。	成功

5. 車を選ぶ

主流：

ーマインメニュではユーザがプレイボタンを押す。

ー車を表示する。

ーユーザが車を選ぶ。

ID	**説明**	**期待結果**	**実際結果**	**ケース結果**

10	前提条件：ユーザがマインメニュ画面でプレイボタンを押して、ほしい車をクリックします。	－正確にユーザの選んだ車を表すことができる。	－正確にユーザの選んだ車を表することができる。	成功
11	前提条件：ユーザがマインメニュ画面でプレイボタンを押して、車を探して、ほしい車をクリックします。	－正確にユーザの選んだ車を表すことができる。	－正確にユーザの選んだ車を表することができる。	成功
12	前提条件：ユーザは車を選んだ。	－ゲームプレイで正しい車を表示する。	－ゲームプレイで正しい車を表示する。	成功

6. 車をアンロック

主流：

－ユーザが車を見て、コインを使用して新しい車をアンロックすることができます。

ID	説明	期待結果	実際結果	ケース結果
13	前提条件：ユーザは車を表す画面がいる。 －コインが足りる場合はアンロックボタンを押す	－新しい車をアンロックすることができる。	－新しい車をアンロックすることができる。	成功
14	前提条件：ユーザは車を表す画面がいる。 －コインが足りな	－新しい車をアンロックすることができる。	－新しい車をアンロックすることができる。	成功

	い場合はアンロックボタンを押す			

7. 車を調節する

主流：

ーゲームプレイスクリーンで車を調節することです。

ID	説明	期待結果	実際結果	ケース結果
15	前提条件：ユーザはゲームをします。 ー右に曲がる	ー車が右に曲がるように運動します。	ー車が右に曲がるように運動します。	成功
16	前提条件：ユーザはゲームをします。 ー左に曲がる	ー車が左に曲がるように運動します。	ー車が左に曲がるように運動します。	成功
17	前提条件：ユーザはゲームをします。 ーブレーキを掛ける	ー車のスピードが下がる。	ー車のスピードが下がる。	成功

8. スピードブースト

主流：

ーゲームプレイスクリーンでブーストボタンを押して、車のスピードがどっと増えている。ボタンを掴まっていれば、エネルギーがないまでスピードが高めて運動する。

ID	説明	期待結果	実際結果	ケース結果
18	前提条件：ユーザはゲームをします。 ーブーストボタンを押す。	ー車のスピードを高める。	ー車のスピードを高める。	成功
19	前提条件：ユーザはゲームをします。 ーエネルギーが少ない場合、ブーストボタンを掴まっている。	ー一時で車のスピードを高める。	ー一時で車のスピードを高める。	成功
20	前提条件：ユーザはゲームをします。 ーエネルギーがない場合、ブーストボタンを掴まっている。	ー車のスピードがままです。	ー車のスピードがままです。	成功

9. アイテムを拾う

主流：

ーゲームプレイスクリーンで道でアイテムを拾って、それぞれブーストを受けることです。

9.1 スルーに運動

ID	説明	期待結果	実際結果	ケース結果

21	前提条件：ユーザはゲームをします。 ースルーに運動アイテムを拾う。	ーユーザの車がほかの車に打って、何もいけない、スルーに運動する。	ーユーザの車がほかの車に打って、何もいけない、スルーに運動する。	成功
22	前提条件：ユーザはゲームをします。 ースルーに運動アイテムを拾う。 ー７秒以内。	ーユーザの車がほかの車に打って、何もいけない、スルーに運動する。	ーユーザの車がほかの車に打って、何もいけない、スルーに運動する。	成功
23	前提条件：ユーザはゲームをします。 ースルーに運動アイテムを拾う。 ー７秒後。	ーユーザの車がほかの車に打って、スルーに運動することができない。	ーユーザの車がほかの車に打って、スルーに運動することができない。	成功

9.2 コインを二倍に受ける

ID	説明	期待結果	実際結果	ケース結果
24	前提条件：ユーザはゲームをします。 ーダブルコインアイテムを拾う。	ーコインを二倍に受け取る。	ーコインを二倍に受け取る。	成功
25	前提条件：ユーザはゲームをします。 ーダブルコインアイテムを拾う。	ーコインを二倍に受け取る。	ーコインを二倍に受け取る。	成功

	－７秒以内。			
26	前提条件：ユーザはゲームをします。 －ダブルコインアイテムを拾う。 －７秒後。	－コインを二倍に受け取らない。	－コインを二倍に受け取らない。	成功

9.3 磁石

ID	**説明**	**期待結果**	**実際結果**	**ケース結果**
27	前提条件：ユーザはゲームをします。 －磁石アイテムを拾う。	－自動にコインを受け取る。	－自動にコインを受け取る。	成功
28	前提条件：ユーザはゲームをします。 －磁石アイテムを拾う。 －７秒以内。	－自動にコインを受け取る。	－自動にコインを受け取る。	成功
29	前提条件：ユーザはゲームをします。 －磁石アイテムを拾う。 －７秒後。	－自動にコインを受け取らない。	－自動にコインを受け取らない。	成功

9.4 バズーカ

ID	説明	期待結果	実際結果	ケース結果
30	前提条件：ユーザはゲームをします。 ーバズーカアイテムを拾う。	ー自動にヘリコプターを打つ。	ー自動にヘリコプターを打つ。	成功

9.5 スーパー

ID	説明	期待結果	実際結果	ケース結果
31	前提条件：ユーザはゲームをします。 ースーパーアイテムを拾う。	ーほかの車を打つ特、影響を及ぼす。	ーほかの車を打つ特、影響を及ぼす。	成功
32	前提条件：ユーザはゲームをします。 ースーパーアイテムを拾う。 ー７秒以内。	ーほかの車を打つ特、影響を及ぼす。	ーほかの車を打つ特、影響を及ぼす。	成功
33	前提条件：ユーザはゲームをします。 ースーパーアイテムを拾う。 ー７秒後。	ーほかの車を打つ特、ゲームオーバー。	ーほかの車を打つ特、ゲームオーバー。	成功

9.6 瞬時エネルギー

ID	説明	期待結果	実際結果	ケース結果
34	前提条件：ユーザはゲームをします。 ー瞬時エネルギーアイテムを拾う。 ーエネルギー量が少ない	ーエネルギーがフルになる。	ーエネルギーがフルになる。	成功
35	前提条件：ユーザはゲームをします。 ー瞬時エネルギーアイテムを拾う。 ーエネルギー量がフルです。	ーエネルギーがフルになる。	ーエネルギーがフルになる。	成功

10. 奏功シェア

主流：

ーゲームプレイ後コインと距離を達成するポイントをシェアすることができる。

ID	説明	期待結果	実際結果	ケース結果
36	前提条件：ユーザはゲームをします。 ーシェアボタンを押す。	ーFACEBOOK でポイントを共有する。 ー新しい近況をアップデートすることです。	ーFACEBOOK でポイントを共有する。 ー新しい近況をアップデートすることです。	成功
37	前提条件：ユーザはゲームをしま	ーFACEBOOK でポイントを共有する。	ーFACEBOOK でポイントを共有す	成功

	す。 ーシェアボタンを押す。 ーインターネット接続がない。	ー新しい近況をアップデートすることができない。	る。 ー新しい近況をアップデートすることができない。	

11. 高得点を表す

主流：

ーマインメニュで HIGHSCORE ボタンを押して、最初からトータルの距離とトータルのコインを受けるポイントを表示して、さらに、一度で最高得点を表すポイントを表すことができる。

ID	説明	期待結果	実際結果	ケース結果
38	前提条件：ユーザはマインメニュスクリーン画面でいる。 ーHIGHSCORE ボタンを押す。 ー最初から（初回でアプリケーションを起動する）	ートータル距離：０ ートータルコイン：０ ー最高得距離：０ ー最高得コイン：０	ートータル距離：０ ートータルコイン：０ ー最高得距離：０ ー最高得コイン：０	成功
39	前提条件：ユーザはマインメニュスクリーン画面でいる。 ーHIGHSCORE ボタンを押す。 ー最初から（初回	ートータル距離：５５０ ートータルコイン：１５５ ー最高得距離：５５０ ー最高得コイン：１５５	ートータル距離：５５０ ートータルコイン：１５５ ー最高得距離：５５０ ー最高得コイン：１５５	成功

	でアプリケーションを起動して、—レースのゲームをする）。達成ポイントは距離：５５０メート、コインは１５５.			
40	前提条件：ユーザはマインメニュスクリーン画面でいる。 －HIGHSCORE ボタンを押す。 －達成ポイントは距離：１００メート、コインは１５.	－トータル距離：６５０ －トータルコイン：１７０ －最高得距離：５５０ －最高得コイン：１５５	－トータル距離：６５０ －トータルコイン：１７０ －最高得距離：６５０ －最高得コイン：１５５	失敗

＊単体テスト

次、ホワイトボックステスト方法で単体テストでゲームコントロールのクラスを実行してきて, ブランチカバレッジ方法を中心して：

1. PlayCarController – OnTriggerEnter(Collider c)

Source Code
```
void OnTriggerEnter( Collider c )
        {
                if( c.tag.Contains("Coin" ) )
                {
                         coinControl coinScript =
c.gameObject.GetComponent<coinControl>() as coinControl ;
                        coinScript.moveToPlayer = true;
                        Destroy( c );
                        GamePlayController.collectedCoinsCounts++;
```

```
                        if(flag_doubleCoins){
                                GamePlayController.collectedCoinsCounts++;
                        }
                }
        }
```

Pre-value variable	**Case 1**	**Case 2**	**Case 3**	**Case 4**
c.tag.Contains(“Coin”)	true	true	false	false
flag_doubleCoin	true	false	true	false
collectedCoinsCount	4	4	4	4
c.gameObject	!= null	!= null	!= null	!= null
Post-value variable				
coinScript.moveToPlayer	true	true	false	false
collectedCoinsCount	6	5	4	4
c.gameObject	null	null	!=null	!=null
Passed/Failed	Passed	Passed	Passed	Passed

Source Code

```
void OnTriggerEnter( Collider c )
        {
                ...
                else if( c.collider.name.Contains("Magnet" ) )
                {
                        SoundController.Static.playmagnetHit();
                        Destroy(c.gameObject);
                        gameObject.transform.Find("particleMagnet").
                        gameObject.SetActive(true);
                        coinControl.isONMagetPower = true;
                        if(switchOnMagnetPower != null)
                           switchOnMagnetPower(null,null);
                        Invoke("EndMagnetPower",magnetPowerTime);
                }
}
```

Pre-value variable	**Case 1**	**Case 2**
c.tag.Contains(“Magnet”)	true	false
isONMagnetPower	false	false
c.gameObject	!=null	!=null
Post-value variable		
isONMagnetPower	true	false
c.gameObject	null	!=null
Passed/Failed	Passed	Passed

Source Code

```
void OnTriggerEnter( Collider c )
        {
                ...
                else if( c.collider.name.Contains("Ghost" ) )
                {
                        Destroy(c.gameObject);
                        foreach(Material carMaterial in carMaterials){
                                carMaterial.shader = Shader.Find("Particles/Additive");
                        }
                        transform.collider.isTrigger = true;
                        Invoke("changeShader1",ghostPowerTime);
                }
         }
```

Pre-value variable	**Case 1**	**Case 2**
c.tag.Contains(“Ghost”)	true	false
carMaterial.shader	!= “Particles/Additive”	!= “Particles/Additive”
transform.collider.isTrigger	false	false
c.gameObject	!=null	!=null

Post-value variable		
carMaterial.shader	“Particles/Additive”	!= “Particles/Additive”
transform.collider.isTrigger	true	false
c.gameObject	null	!=null
Passed/Failed	Passed	Passed

Source Code

```
void OnTriggerEnter( Collider c )
        {
                ...
                else if( c.collider.name.Contains("DoubleCoins" ) )
                {
                        SoundController.Static.playdoubleHit();
                        Destroy(c.gameObject);
                        gameObject.transform.Find("particleDouble").
                        gameObject.SetActive(true);
                        flag_doubleCoins = true;
                        Invoke("EndDoublePower",doublePowerTime);
                }
         }
```

Pre-value variable	**Case 1**	**Case 2**
c.tag.Contains(“DoubleCoins”)	true	false
gameObject.transform.Find("particleDouble"). gameObject.Active	false	false
flag_doubleCoins	false	false
c.gameObject	!=null	!=null
Post-value variable		
gameObject.transform.Find("particleDouble"). gameObject.Active	true	false

flag_doubleCoins	true	false
c.gameObject	null	!=null
Passed/Failed	Passed	Passed

Source Code

```
void OnTriggerEnter( Collider c )
        {
                ...
                else if( c.collider.name.Contains("InstantNitrous" ) )
                {
                        SoundController.Static.playinstantNitrousHit();
                        Destroy(c.gameObject);
                        NitrousIndicator.NitrousCount=100;
                        NitrousIndicator.Static.UpdateNitrousDisplay();
                }
        }
```

Pre-value variable	**Case 1**	**Case 2**	**Case 3**
c.tag.Contains(“InstantNitrous”)	true	true	false
NitrousIndicator.NitrousCount	0	100	0
c.gameObject	!=null	!=null	!=null
Post-value variable			
NitrousIndicator.NitrousCount	100	100	0
c.gameObject	null	null	!=null
Passed/Failed	Passed	Passed	Passed

Source Code

```
void OnTriggerEnter( Collider c )
        {
                ...
```

```
                    else if (c.collider.name.Contains ("Bazooka")) {

                           Instantiate(missileFighBack, gameObject.transform.position,
                                                             Quaternion.identity);                    Destroy
                                                             (c.gameObject);

                           hitcount += 1;
                    }
  }
```

Pre-value variable	Case 1	Case 2
c.tag.Contains("Bazooka")	true	false
hitCount	0	0
c.gameObject	!=null	!=null
Post-value variable		
hitCount	1	0
c.gameObject	null	!=null
Passed/Failed	Passed	Passed

2. PlayCarController - OnCollisionEnter(Collision incomingCollision)

Source Code

```
void OnCollisionEnter(Collision incomingCollision)
          {
                    string incTag = incomingCollision.collider.tag;
                    if (incTag.Contains ("TrafficCar"))
                    {
                              carSpeed=0;
                              wheelSpeed=0;
                              turnSpeed=0;
                              rigidbody.velocity = Vector3.zero;
                              GamePlayController.isGameEnded = true;
                              if(gameEnded != null) gameEnded(null,null);
                              iTween.ShakePosition(Camera.main.gameObject,new
Vector3(1,1,1),0.6f);
```

```
                        rigidbody.constraints = RigidbodyConstraints.FreezeAll;
                        GameObject trafficCar  = incomingCollision.collider.gameObject ;
                        iTween.RotateTo(trafficCar ,
                        new  Vector3(0,UnityEngine.Random.Range(-1,2)*25,0),1.0f);
                }
                                ...
}
```

Pre-value variable	**Case 1**	**Case 2**	**Case 3**	**Case 4**
incTag.Contains("TrafficCar")	true	true	false	false
carSpeed	>0	>0	>0	>0
wheelSpeed	>0	>0	>0	>0
rigidbody.velocity	!=Vector3.zero	!=Vector3.zero	!=Vector3.zero	!=Vector3.zero
isGameEnded	false	false	false	false
gameEnded	!=null	null	!=null	null
Post-value variable				
carSpeed	0	0	>0	>0
wheelSpeed	0	0	>0	>0
rigidbody.velocity	Vector3.zero	Vector3.zero	!= Vector3.zero	!= Vector3.zero
isGameEnded	true	true	false	false
gameEnded	null	null	!=null	null
Passed/Failed	Passed	Passed	Passed	Passed

Source Code

void OnCollisionEnter(Collision incomingCollision)
        {
 	
                string incTag = incomingCollision.collider.tag;

```
...
                    if ((incTag.Contains ("missile")) ||  (incTag.Contains ("DeadZone"))){

                              carSpeed=0;
                              wheelSpeed=0;
                              isDoubleSpeed=0;
                              turnSpeed=0;
                              rigidbody.velocity = Vector3.zero;
                              isDoubleSpeed = 1;
                              GamePlayController.isGameEnded = true;

                              if(gameEnded != null) gameEnded(null,null);
                                    iTween.ShakePosition(Camera.main.gameObject,

                                                      new  Vector3(1,1,1),0.6f);

                              rigidbody.constraints = RigidbodyConstraints.FreezeAll;
                              GameObject trafficCar  = incomingCollision.collider.gameObject ;
                              trafficCar.SendMessage("StopCar",
SendMessageOptions.DontRequireReceiver);|
                               iTween.RotateTo(trafficCar , new
Vector3(0,UnityEngine.Random.Range(-1,2)*25,0),1.0f);

                               GameObject MissileDes =
                               GameObject.FindGameObjectWithTag("MissileDes")

                                          as GameObject;

                              Destroy(MissileDes);

                              Instantiate(expos,gameObject.transform.position,  Quaternion.identity);

                              Destroy(gameObject);
                    }
}
```

Pre-value variable	**Case 1**	**Case 2**	**Case 3**	**Case 4**
incTag.Contains(“TrafficCar”)	true	true	false	false
carSpeed	>0	>0	>0	>0
wheelSpeed	>0	>0	>0	>0

rigidbody.velocity	!=Vector3.zero	!=Vector3.zero	!=Vector3.zero	!=Vector3.zero
MissileDes	!=null	!=null	!=null	!=null
isGameEnded	false	false	false	false
gameEnded	!=null	null	!=null	null
Post-value variable				
carSpeed	0	0	>0	>0
wheelSpeed	0	0	>0	>0
rigidbody.velocity	Vector3.zero	Vector3.zero	!= Vector3.zero	!= Vector3.zero
MissileDes	null	null	null	null
isGameEnded	true	true	false	false
gameEnded	null	null	!=null	null
Passed/Failed	Passed	Passed	Passed	Passed

3. RoadController – OnTriggerEnter(Collider c)

Source Code

```
                    if( c.tag.Contains("Player") && justOnce==false)
                    {                   GameObject newBlock =   otherRoadBlock
newBlock.name="road" + roadBlockCount;
                            roadBlockCount++;
                            otherRoadBlock.transform.Translate( 0,0, 2782.907f*2);
                            justOnce=true;
                            Destroy(this.gameObject,20);
                            Invoke("SwitchRoadBlocks",4 );
                     }
```

Pre-value variable	**Case 1**	**Case 2**	**Case 3**	**Case 4**
c.tag.Contains(“Player”)	true	false	true	false
justOnce	false	false	true	true

newBlock.name	"road0"	"road0"	"road0"	"road0"
otherRoadBlock.transform.position	(1,1,1000)	(1,1,1000)	(1,1,1000)	(1,1,1000)
Post-value variable				
justOnce	true	false	false	false
newBlock.name	"road1"	"road0"	"road0"	"road0"
otherRoadBlock.transform.position	(1,1,6565.814)	(1,1,1000)	(1,1,1000)	(1,1,1000)
Passed/Failed	Passed	Passed	Passed	Passed

4. HelicopterController – OnTriggerEnter(Collider c)

Source Code

```
void OnCollisionEnter (Collision c)
{
        if (c.gameObject.tag == " BazookaMissile ")
        {
                gameplaycontrol.CancelFire();
                Destroy (this.gameObject);
         }

}
```

Pre-value variable	**Case 1**	**Case 2**
c.tag.Contains("BazookaMissile")	true	false
gameObject	!=null	!=null
Post-value variable		
gameObject	null	!=null
Passed/Failed	Passed	Passed

5. CoinController.CS

ID	ライン#	条件	TRUE		FALSE		ケース結果
			期待	実際	期待	実際	
1	27	(isONMagetPower)	onMagetPower = true;	onMagetPower = true;	onMagetPower = false;	onMagetPower = false;	成功
2	47	(box != null)	box.size = new Vector3 (9, 9, 9);	box.size = new Vector3 (9, 9, 9);	box.size = new Vector3 (1, 1, 1);	box.size = new Vector3 (1, 1, 1);	成功
3	72	(moveToPlayer)	thisTrans.position = Vector3.MoveTowards (thisTrans.position, playerCarControl.thisPosition , 2.0f*playerCarControl.isDoubleSpeed);	thisTrans.position = Vector3.MoveTowards (thisTrans.position, playerCarControl.thisPosition , 2.0f*playerCarControl.isDoubleSpeed);	thisTrans.position = transform	thisTrans.position = transform	成功
4	74	(thisTrans.position.z < playerCarControl.thisPosition.z)	moveToPlayer=false;	moveToPlayer=false;			成功
5	74	(thisTrans.position.z = playerCarControl.thisPosition.z)	moveToPlayer=true;	moveToPlayer=true;			成功
6	74	(thisTrans.position.z > playerCarControl.this	moveToPlayer=true;	moveToPlayer=true;			成功

		Position.z)					
7	76	NitrousIndicator.NitrousCount = 90	NitrousIndicator.NitrousCount+=1.0f;	NitrousIndicator.NitrousCount+=1.0f;			成功
8	76	NitrousIndicator.NitrousCount = 99	NitrousIndicator.NitrousCount+=1.0f;	NitrousIndicator.NitrousCount+=1.0f;			成功
9	76	NitrousIndicator.NitrousCount = 101			NitrousIndicator.NitrousCoun= NitrousIndicator.NitrousCoun	NitrousIndicator.NitrousCoun= NitrousIndicator.NitrousCoun	成功
10	76	NitrousIndicator.NitrousCount = 0	NitrousIndicator.NitrousCount+=1.0f;	NitrousIndicator.NitrousCount+=1.0f;			成功
11	76	NitrousIndicator.NitrousCount = 999			NitrousIndicator.NitrousCoun= NitrousIndicator.NitrousCoun	NitrousIndicator.NitrousCoun= NitrousIndicator.NitrousCoun	成功
12	76	NitrousIndicator.NitrousCount = 9999			NitrousIndicator.NitrousCoun= NitrousIndicator.NitrousCoun	NitrousIndicator.NitrousCoun= NitrousIndicator.NitrousCoun	成功
13	76	NitrousIndicator.NitrousCount = 99999			NitrousIndicator.NitrousCoun= NitrousIndicator.NitrousCoun	NitrousIndicator.NitrousCoun= NitrousIndicator.NitrousCoun	成功

6. GamePlayController.CS

ID	ライン#	条件	TRUE		FALSE		ケース結果
			期待	実際	期待	実際	
14	79	(DroneChase!= null)	InvokeRepeating("generateBazooka",13,10); InvokeRepeating ("generateFightArea2", 13, 4);	InvokeRepeating("generateBazooka",13,10); InvokeRepeating ("generateFightArea2", 13, 4);	Generate nothing.	Generate nothing.	成功
15	97	(camScript == null)	camScript = Camera.main.GetComponent<carCamera>();	camScript = Camera.main.GetComponent<carCamera>();			成功
16	111	(isGameEnded)	coinsText.text ="" ;distanceText.text="" ;gameOverTxt.text = "GAME OVER :(";	coinsText.text ="" ;distanceText.text="" ;gameOverTxt.text = "GAME OVER :(";	distanceTravelled = Mathf.RoundToInt ((playerCarControl.thisPosition.z + (1104.015f))/ 10);	distanceTravelled = Mathf.RoundToInt ((playerCarControl.thisPosition.z + (1104.015f))/ 10);	成功
17	200	(GameData.ExperientOnRun < GameData.CurrentLevel * GameData.ExperientLevelBegin * (1 - GameData.PercentLevel))	GameData.PercentLevel = GameData.PercentLevel + ((float)GameData.ExperientOnRun/ (float)(Game	GameData.PercentLevel = GameData.PercentLevel + ((float)GameData.ExperientOnRun/ (float)(Game	GameData.CurrentLevel += 1; GameData.PercentLevel = 0;	GameData.CurrentLevel += 1; GameData.PercentLevel = 0;	成功

			Data.CurrentLevel * GameData.ExperientLevelBegin))	Data.CurrentLevel * GameData.ExperientLevelBegin))			

7. PlayerCarController.CS

ID	ライン#	条件	TRUE		FALSE		ケース結果
			期待	実際	期待	実際	
18	63	#if UNITY_WEBPLAYER || UNITY_EDITOR	tilt = tilt*2;	tilt = tilt*2;	tilt = tilt;	tilt = tilt;	成功
19	70	(t.name.Contains("Effect"))	particleParent = t.gameObject;	particleParent = t.gameObject;			成功
20	111	(isGameEnded)	coinsText.text ="" ;distanceText.text="" ;gameOverTxt.text = "GAME OVER :(";	coinsText.text ="" ;distanceText.text="" ;gameOverTxt.text = "GAME OVER :(";	distanceTravelled = Mathf.RoundToInt ((playerCarControl.thisPosition.z + (1104.015f))/ 10);	distanceTravelled = Mathf.RoundToInt ((playerCarControl.thisPosition.z + (1104.015f))/ 10);	成功
21	103	(isDoubleSpeed == 1)	hideNitrousParticle();	hideNitrousParticle();	showNitrousParticle ();	showNitrousParticle ();	成功
22	109	(PercentSpeed < MaxPercentSpeed)	carSpeed = OriginalCarSpeed*PercentSpeed/100;	carSpeed = OriginalCarSpeed*PercentSpeed/100;			成功

23	109	(PercentSpeed = MaxPercentSpeed)	carSpeed = OriginalCarSpeed;	carSpeed = OriginalCarSpeed;			成功
24	109	(PercentSpeed >MaxPercentSpeed)	carSpeed = OriginalCarSpeed;	carSpeed = OriginalCarSpeed;			成功
25	118	(c.tag.Contains("coin")) - c.tag = coin	coinControl coinScript = c.gameObject.GetComponent<coinControl>() as coinControl ; coinScript.moveToPlayer = true; Destroy(c); GamePlayController.collectedCoinsCounts ++;	coinControl coinScript = c.gameObject.GetComponent<coinControl>() as coinControl ; coinScript.moveToPlayer = true; Destroy(c); GamePlayController.collectedCoinsCounts ++;			成功
26	118	(c.tag.Contains("coin")) - c.tag = coinnn	coinControl coinScript = c.gameObject.GetComponent<coinControl>() as coinControl ; coinScript.moveToPlayer = true; Destroy(c); GamePlayController.collectedCoinsCounts ++;	coinControl coinScript = c.gameObject.GetComponent<coinControl>() as coinControl ; coinScript.moveToPlayer = true; Destroy(c); GamePlayController.collectedCoinsCounts ++;			成功
27	118	(c.tag.Contains("coin"))	coinControl coinScript =	coinControl coinScript =	Nothing happens. C is	Nothing happens. C is	成功

		- c.tag = ccccoin	c.gameObject.GetComponent<coinControl>() as coinControl ; coinScript.moveToPlayer = true; Destroy(c); GamePlayController.collectedCoinsCounts ++;	c.gameObject.GetComponent<coinControl>() as coinControl ; coinScript.moveToPlayer = true; Destroy(c); GamePlayController.collectedCoinsCounts ++;	not destroyed.	not destroyed.	
28	118	(c.tag.Contains("coin")) - c.tag = cooin			Nothing happens. C is not destroyed.	Nothing happens. C is not destroyed.	成功
29	118	(c.tag.Contains("coin")) - c.tag = coiin			Nothing happens. C is not destroyed.	Nothing happens. C is not destroyed.	成功
30	118	(c.tag.Contains("coin")) - c.tag = cooiin			Nothing happens. C is not destroyed.	Nothing happens. C is not destroyed.	成功
31	118	(c.tag.Contains("coin")) - c.tag = ccooiinn			Nothing happens. C is not destroyed	Nothing happens. C is not destroyed	成功
32	124	(flag_doubleCoins)	GamePlayController.collectedCoinsCounts ++;	GamePlayController.collectedCoinsCounts ++;	GamePlayController.collectedCoinsCounts = GamePlayController.collectedCoinsCounts ;	GamePlayController.collectedCoinsCounts = GamePlayController.collectedCoinsCounts ;	成功
3	129	(c.collider.name.Con	SoundControl	SoundControl	Nothing	Nothing	成功

3		tains("Magnet")) c.collider.name = Magnet	ler.Static.playmagnetHit(); Destroy(c.gameObject); gameObject.transform.Find("particleMagnet").gameObject.SetActive(true); coinControl.isONMagetPower = true; if(switchOnMagnetPower != null) switchOnMagnetPower(null,null); Invoke("EndMagnetPower",magnetPowerTime);	ler.Static.playmagnetHit(); Destroy(c.gameObject); gameObject.transform.Find("particleMagnet").gameObject.SetActive(true); coinControl.isONMagetPower = true; if(switchOnMagnetPower != null) switchOnMagnetPower(null,null); Invoke("EndMagnetPower",magnetPowerTime);	happens.	happens.	
34	129	(c.collider.name.Contains("Magnet")) c.collider.name = Magnett	SoundController.Static.playmagnetHit(); Destroy(c.gameObject); gameObject.transform.Find("particleMagnet").gameObject.SetActive(true); coinControl.isONMagetPower = true; if(switchOnMagnetPower != null)	SoundController.Static.playmagnetHit(); Destroy(c.gameObject); gameObject.transform.Find("particleMagnet").gameObject.SetActive(true); coinControl.isONMagetPower = true; if(switchOnMagnetPower != null)			成功

			switchOnMagnetPower(null,null); Invoke("EndMagnetPower",magnetPowerTime);	switchOnMagnetPower(null,null); Invoke("EndMagnetPower",magnetPowerTime);			
35	129	(c.collider.name.Contains("Magnet")) c.collider.name = MMagnet	SoundController.Static.playmagnetHit(); Destroy(c.gameObject); gameObject.transform.Find("particleMagnet").gameObject.SetActive(true); coinControl.isONMagetPower = true; if(switchOnMagnetPower != null) switchOnMagnetPower(null,null); Invoke("EndMagnetPower",magnetPowerTime);	SoundController.Static.playmagnetHit(); Destroy(c.gameObject); gameObject.transform.Find("particleMagnet").gameObject.SetActive(true); coinControl.isONMagetPower = true; if(switchOnMagnetPower != null) switchOnMagnetPower(null,null); Invoke("EndMagnetPower",magnetPowerTime);			成功
36	129	(c.collider.name.Contains("Magnet")) c.collider.name = Maagnet			Nothing happens.	Nothing happens.	成功
37	129	(c.collider.name.Contains("Magnet")) c.collider.name =			Nothing happens.	Nothing happens.	成功

		Maggnet					
38	129	(c.collider.name.Contains("Magnet")) c.collider.name = Magnnet			Nothing happens.	Nothing happens.	成功
39	129	(c.collider.name.Contains("Magnet")) c.collider.name = Magneet			Nothing happens.	Nothing happens.	成功
40	139	(c.collider.name.Contains("Ghost")) c.collider.name = Ghost	Destroy(c.gameObject); foreach(Material carMaterial in carMaterials){ carMaterial.shader = Shader.Find("Particles/Additive"); } transform.collider.isTrigger = true; Invoke("changeShader1",ghostPowerTime);	Destroy(c.gameObject); foreach(Material carMaterial in carMaterials){ carMaterial.shader = Shader.Find("Particles/Additive"); } transform.collider.isTrigger = true; Invoke("changeShader1",ghostPowerTime);			成功
41	139	(c.collider.name.Contains("Ghost")) c.collider.name = Ghostt	Destroy(c.gameObject); foreach(Material carMaterial in carMaterials){ carMaterial.sh	Destroy(c.gameObject); foreach(Material carMaterial in carMaterials){ carMaterial.sh			成功

			ader = Shader.Find("Particles/Additive"); } transform.collider.isTrigger = true; Invoke("changeShader1",ghostPowerTime);	ader = Shader.Find("Particles/Additive"); } transform.collider.isTrigger = true; Invoke("changeShader1",ghostPowerTime);			

42	139	(c.collider.name.Contains("Ghost"))
		
		c.collider.name = GGhost
			Destroy(c.gameObject);
			
			foreach(Material carMaterial in carMaterials){
			
			carMaterial.shader = Shader.Find("Particles/Additive"); } transform.collider.isTrigger = true; Invoke("changeShader1",ghostPowerTime);
				Destroy(c.gameObject);
				
				foreach(Material carMaterial in carMaterials){
				
				carMaterial.shader = Shader.Find("Particles/Additive"); } transform.collider.isTrigger = true; Invoke("changeShader1",ghostPowerTime);
							成功

43	139	(c.collider.name.Contains("Ghost")) c.collider.name = Ghhost			Nothing happens.	Nothing happens.	成功
44	139	(c.collider.name.Contains("Ghost")) c.collider.name =			Nothing happens.	Nothing happens.	成功

		Ghoost					
45	139	(c.collider.name.Contains("Ghost")) c.collider.name = Ghosst			Nothing happens.	Nothing happens.	成功
46	139	(c.collider.name.Contains("Ghost")) c.collider.name = GGhhost			Nothing happens.	Nothing happens.	成功
47	151	(c.collider.name.Contains("DoubleCoins")) c.collider.name = DoubleCoins	SoundController.Static.playdoubleHit(); Destroy(c.gameObject); gameObject.transform.Find("particleDouble").gameObject.SetActive(true); flag_doubleCoins = true; Invoke("EndDoublePower",doublePowerTime);	SoundController.Static.playdoubleHit(); Destroy(c.gameObject); gameObject.transform.Find("particleDouble").gameObject.SetActive(true); flag_doubleCoins = true; Invoke("EndDoublePower",doublePowerTime);			成功
48	151	(c.collider.name.Contains("DoubleCoins")) c.collider.name = DoubleCoinss	SoundController.Static.playdoubleHit(); Destroy(c.gameObject); gameObject.transform.Find("particleDouble").gameObject.SetActive(true);	SoundController.Static.playdoubleHit(); Destroy(c.gameObject); gameObject.transform.Find("particleDouble").gameObject.SetActive(true);			成功

			flag_doubleCoins = true; Invoke("EndDoublePower",doublePowerTime);	flag_doubleCoins = true; Invoke("EndDoublePower",doublePowerTime);			
49	151	(c.collider.name.Contains("DoubleCoins")) c.collider.name = DDoubleCoins	SoundController.Static.playdoubleHit(); Destroy(c.gameObject); gameObject.transform.Find("particleDouble").gameObject.SetActive(true); flag_doubleCoins = true; Invoke("EndDoublePower",doublePowerTime);	SoundController.Static.playdoubleHit(); Destroy(c.gameObject); gameObject.transform.Find("particleDouble").gameObject.SetActive(true); flag_doubleCoins = true; Invoke("EndDoublePower",doublePowerTime);			成功
50	151	(c.collider.name.Contains("DoubleCoins")) c.collider.name = DooubleCoins			Nothing happens	Nothing happens	成功
51	151	(c.collider.name.Contains("DoubleCoins")) c.collider.name = DouubleCoins			Nothing happens	Nothing happens	成功
52	151	(c.collider.name.Contains("DoubleCoins"))			Nothing happens	Nothing happens	成功

		c.collider.name = DoubbleCoins					
53	151	(c.collider.name.Contains("DoubleCoins")) c.collider.name = DoublleCoins			Nothing happens	Nothing happens	成功
54	160	(c.collider.name.Contains("InstantNitrous")) c.collider.name = InstantNitrous	SoundController.Static.playinstantNitrousHit(); Destroy(c.gameObject); NitrousIndicator.NitrousCount=100; NitrousIndicator.Static.UpdateNitrousDisplay();	SoundController.Static.playinstantNitrousHit(); Destroy(c.gameObject); NitrousIndicator.NitrousCount=100; NitrousIndicator.Static.UpdateNitrousDisplay();			成功
55	160	(c.collider.name.Contains("InstantNitrous")) c.collider.name = InstantNitrouss	SoundController.Static.playinstantNitrousHit(); Destroy(c.gameObject); NitrousIndicator.NitrousCount=100; NitrousIndicator.Static.UpdateNitrousDisplay();	SoundController.Static.playinstantNitrousHit(); Destroy(c.gameObject); NitrousIndicator.NitrousCount=100; NitrousIndicator.Static.UpdateNitrousDisplay();			成功
56	160	(c.collider.name.Contains("InstantNitrous")) c.collider.name =	SoundController.Static.playinstantNitrousHit(); Destroy(c.ga	SoundController.Static.playinstantNitrousHit(); Destroy(c.ga			成功

		IInstantNitrous	meObject); NitrousIndicator.NitrousCount=100; NitrousIndicator.Static.UpdateNitrousDisplay();	meObject); NitrousIndicator.NitrousCount=100; NitrousIndicator.Static.UpdateNitrousDisplay();			
57	160	(c.collider.name.Contains("InstantNitrous")) c.collider.name = InstantNNitrous			Nothing happens	Nothing happens	成功
58	160	(c.collider.name.Contains("InstantNitrous")) c.collider.name = IInstantNitrous			Nothing happens	Nothing happens	成功
59	160	(c.collider.name.Contains("InstantNitrous")) c.collider.name = InstanttNitrous			Nothing happens	Nothing happens	成功
60	160	(c.collider.name.Contains("InstantNitrous")) c.collider.name = InstanntNitrous			Nothing happens	Nothing happens	成功
61	174	(c.collider.name.Contains ("bazoka")) c.collider.name = bazoka	Instantiate(missileFighBack,gameObject.transform.position, Quaternion.identity);	Instantiate(missileFighBack,gameObject.transform.position, Quaternion.identity);			成功

			Destroy (c.gameObject); hitcount += 1;	Destroy (c.gameObject); hitcount += 1;			
62	174	(c.collider.name.Contains ("bazoka")) c.collider.name = bbazoka	Instantiate(missileFighBack,gameObject.transform.position, Quaternion.identity); Destroy (c.gameObject); hitcount += 1;	Instantiate(missileFighBack,gameObject.transform.position, Quaternion.identity); Destroy (c.gameObject); hitcount += 1;			成功
63	174	(c.collider.name.Contains ("bazoka")) c.collider.name = bazokaa	Instantiate(missileFighBack,gameObject.transform.position, Quaternion.identity); Destroy (c.gameObject); hitcount += 1;	Instantiate(missileFighBack,gameObject.transform.position, Quaternion.identity); Destroy (c.gameObject); hitcount += 1;			成功
64	174	(c.collider.name.Contains ("bazoka")) c.collider.name = baazoka			Nothing happens	Nothing happens	成功
65	174	(c.collider.name.Contains ("bazoka")) c.collider.name = bazzoka			Nothing happens	Nothing happens	成功
66	174	(c.collider.name.Contains ("bazoka"))			Nothing happens	Nothing happens	成功

		c.collider.name = bazooka					
67	174	(c.collider.name.Contains ("bazoka")) c.collider.name = bazokka			Nothing happens	Nothing happens	成功
68	219	(count_shader <= 3) count_shader = 3	Invoke ("changeShader1", 0.2f);	Invoke ("changeShader1", 0.2f);			成功
69	219	(count_shader <= 3) count_shader = 2	Invoke ("changeShader1", 0.2f);	Invoke ("changeShader1", 0.2f);			成功
70	219	(count_shader <= 3) count_shader = 4			count_shader = 0;	count_shader = 0;	成功
71	219	(count_shader <= 3) count_shader = 5			count_shader = 0;	count_shader = 0;	成功
72	219	(count_shader <= 3) count_shader = 9			count_shader = 0;	count_shader = 0;	成功
73	219	(count_shader <= 3) count_shader = 99			count_shader = 0;	count_shader = 0;	成功
74	219	(count_shader <= 3) count_shader = 999			count_shader = 0;	count_shader = 0;	成功
75	231	(incTag.Contains ("trafficCar")) incTag = trafficCar	GameObject trafficCar = incomingCollision.collider.gameObject ; trafficCar.SendMessage("StopCar",SendMessageOptio	GameObject trafficCar = incomingCollision.collider.gameObject ; trafficCar.SendMessage("StopCar",SendMessageOptio			成功

			ns.DontRequireReceiver); iTween.RotateTo(trafficCar , new Vector3(0,UnityEngine.Random.Range(-1,2)*25,0),1.0f);	ns.DontRequireReceiver); iTween.RotateTo(trafficCar , new Vector3(0,UnityEngine.Random.Range(-1,2)*25,0),1.0f);			
76	231	(incTag.Contains ("trafficCar")) incTag = trafficCarr	GameObject trafficCar = incomingCollision.collider.gameObject ; trafficCar.SendMessage("StopCar",SendMessageOptions.DontRequireReceiver); iTween.RotateTo(trafficCar , new Vector3(0,UnityEngine.Random.Range(-1,2)*25,0),1.0f);	GameObject trafficCar = incomingCollision.collider.gameObject ; trafficCar.SendMessage("StopCar",SendMessageOptions.DontRequireReceiver); iTween.RotateTo(trafficCar , new Vector3(0,UnityEngine.Random.Range(-1,2)*25,0),1.0f);			成功
77	231	(incTag.Contains ("trafficCar")) incTag = ttrafficCar	GameObject trafficCar = incomingCollision.collider.gameObject ; trafficCar.SendMessage("StopCar",SendMessageOptions.DontRequireReceiver);	GameObject trafficCar = incomingCollision.collider.gameObject ; trafficCar.SendMessage("StopCar",SendMessageOptions.DontRequireReceiver);			成功

			iTween.RotateTo(trafficCar , new Vector3(0,UnityEngine.Random.Range(-1,2)*25,0),1.0f);	iTween.RotateTo(trafficCar , new Vector3(0,UnityEngine.Random.Range(-1,2)*25,0),1.0f);			
78	231	(incTag.Contains ("trafficCar")) incTag = traffficCar			Nothing happen when OnCollisionEnter	Nothing happen when OnCollisionEnter	成功
79	231	(incTag.Contains ("trafficCar")) incTag = traffiicCar			Nothing happen when OnCollisionEnter	Nothing happen when OnCollisionEnter	成功
80	231	(incTag.Contains ("trafficCar")) incTag = trafficcCar			Nothing happen when OnCollisionEnter	Nothing happen when OnCollisionEnter	成功
81	231	(incTag.Contains ("trafficCar")) incTag = trafficCCar			Nothing happen when OnCollisionEnter	Nothing happen when OnCollisionEnter	成功
82	251	(incTag.Contains ("missile")) incTag = missle	GameObject MissileDes = GameObject.FindGameObjectWithTag("MissileDes") as GameObject; Destroy(MissileDes); Instantiate(expos,gameObject.transform.position,	GameObject MissileDes = GameObject.FindGameObjectWithTag("MissileDes") as GameObject; Destroy(MissileDes); Instantiate(expos,gameObject.transform.position,			成功

			Quaternion.identity); Destroy(gameObject);	Quaternion.identity); Destroy(gameObject);			
83	251	(incTag.Contains ("missile")) incTag = misslee	GameObject MissileDes = GameObject. FindGameObjectWithTag(" MissileDes") as GameObject; Destroy(MissileDes); Instantiate(expos,gameObject.transform.position, Quaternion.identity); Destroy(gameObject);	GameObject MissileDes = GameObject. FindGameObjectWithTag(" MissileDes") as GameObject; Destroy(MissileDes); Instantiate(expos,gameObject.transform.position, Quaternion.identity); Destroy(gameObject);			
84	251	(incTag.Contains ("missile")) incTag = mmissle	GameObject MissileDes = GameObject. FindGameObjectWithTag(" MissileDes") as GameObject; Destroy(MissileDes); Instantiate(expos,gameObject.transform.position, Quaternion.identity); Destroy(game	GameObject MissileDes = GameObject. FindGameObjectWithTag(" MissileDes") as GameObject; Destroy(MissileDes); Instantiate(expos,gameObject.transform.position, Quaternion.identity); Destroy(game			成功

			Object);	Object);			
85	251	(incTag.Contains ("missile")) incTag = miissle			Nothing happen when OnCollisionEnter	Nothing happen when OnCollisionEnter	成功
86	251	(incTag.Contains ("missile")) incTag = misssle			Nothing happen when OnCollisionEnter	Nothing happen when OnCollisionEnter	成功
87	251	(incTag.Contains ("missile")) incTag = misslle			Nothing happen when OnCollisionEnter	Nothing happen when OnCollisionEnter	成功
88	251	(incTag.Contains ("missile")) incTag = missslle			Nothing happen when OnCollisionEnter	Nothing happen when OnCollisionEnter	成功

＊マルチプレイヤーで単体テスト

ブラックボックステストを中心する方法で機能テストを実行する。

1. 車を調節する

主流：

ーゲームプレイスクリーンで車を調節することです。

ID	説明	期待結果	実際結果	ケース結果
1	前提条件：ユーザはゲームをします。 ー右に曲がる	ー車が右に曲がるように運動します。	ー車が右に曲がるように運動します。	成功

2	前提条件：ユーザはゲームをします。 ー左に曲がる	ー車が左に曲がるように運動します。	ー車が左に曲がるように運動します。	成功
3	前提条件：ユーザはゲームをします。 ーブレーキを掛ける	ー車のスピードが下がる。	ー車のスピードが下がる。	成功

2. スピードブースト

主流：

ーゲームプレイスクリーンでブーストボタンを押して、車のスピードがどっと増えている。ボタンを掴まっていれば、エネルギーがないまでスピードが高めて運動する。

ID	説明	期待結果	実際結果	ケース結果
4	前提条件：ユーザはゲームをします。 ーブーストボタンを押す。	ー車のスピードを高める。	ー車のスピードを高める。	成功
5	前提条件：ユーザはゲームをします。 ーエネルギーが少ない場合、ブーストボタンを掴まっている。	ー一時で車のスピードを高める。	ー一時で車のスピードを高める。	成功

6	前提条件：ユーザはゲームをします。 ーエネルギーがない場合、ブーストボタンを掴まっている。	ー車のスピードがままです。	ー車のスピードがままです。	成功

3. アイテムを拾う

主流：

ーゲームプレイスクリーンで道でアイテムを拾って、それぞれブーストを受けることです。

3.1 スルーに運動

ID	説明	期待結果	実際結果	ケース結果
7	前提条件：ユーザはゲームをします。 ースルーに運動アイテムを拾う。	ーユーザの車がほかの車に打って、何もいけない、スルーに運動する。	ーユーザの車がほかの車に打って、何もいけない、スルーに運動する。	成功
8	前提条件：ユーザはゲームをします。 ースルーに運動アイテムを拾う。 ー７秒以内。	ーユーザの車がほかの車に打って、何もいけない、スルーに運動する。	ーユーザの車がほかの車に打って、何もいけない、スルーに運動する。	成功
9	前提条件：ユーザはゲームをします。	ーユーザの車がほかの車に打って、スルーに運動することが	ーユーザの車がほかの車に打って、スルーに運動する	成功

	ースルーに運動アイテムを拾う。 ー７秒後。	できない。	ことができない。	

3.2 コインを二倍に受ける

ID	説明	期待結果	実際結果	ケース結果
10	前提条件：ユーザはゲームをします。 ーダブルコインアイテムを拾う。	ーコインを二倍に受け取る。	ーコインを二倍に受け取る。	成功
11	前提条件：ユーザはゲームをします。 ーダブルコインアイテムを拾う。 ー７秒以内。	ーコインを二倍に受け取る。	ーコインを二倍に受け取る。	成功
12	前提条件：ユーザはゲームをします。 ーダブルコインアイテムを拾う。 ー７秒後。	ーコインを二倍に受け取らない。	ーコインを二倍に受け取らない。	成功

3.3 磁石

ID	説明	期待結果	実際結果	ケース結果
13	前提条件：ユーザはゲームをします。	ー自動にコインを受け取る。	ー自動にコインを受け取る。	成功

	ー磁石アイテムを拾う。			
14	前提条件：ユーザはゲームをします。 ー磁石アイテムを拾う。 ー７秒以内。	ー自動にコインを受け取る。	ー自動にコインを受け取る。	成功
15	前提条件：ユーザはゲームをします。 ー磁石アイテムを拾う。 ー７秒後。	ー自動にコインを受け取らない。	ー自動にコインを受け取らない。	成功

3.4 バズーカ

ID	説明	期待結果	実際結果	ケース結果
16	前提条件：ユーザはゲームをします。 ーバズーカアイテムを拾う。	ー自動にヘリコプターを打つ。	ー自動にヘリコプターを打つ。	成功

3.5 スーパー

ID	説明	期待結果	実際結果	ケース結果
17	前提条件：ユーザはゲームをします。 ースーパーアイテムを拾う。	ーほかの車を打つ特、影響を及ぼす。	ーほかの車を打つ特、影響を及ぼす。	成功

18	前提条件：ユーザはゲームをします。 ースーパーアイテムを拾う。 ー７秒以内。	ーほかの車を打つ特、影響を及ぼす。	ーほかの車を打つ特、影響を及ぼす。	成功
19	前提条件：ユーザはゲームをします。 ースーパーアイテムを拾う。 ー７秒後。	ーほかの車を打つ特、ゲームオーバー。	ーほかの車を打つ特、ゲームオーバー。	成功

3.6 瞬時エネルギー

ID	説明	期待結果	実際結果	ケース結果
20	前提条件：ユーザはゲームをします。 ー瞬時エネルギーアイテムを拾う。 ーエネルギー量が少ない	ーエネルギーがフルになる。	ーエネルギーがフルになる。	成功
21	前提条件：ユーザはゲームをします。 ー瞬時エネルギーアイテムを拾う。 ーエネルギー量がフルです。	ーエネルギーがフルになる。	ーエネルギーがフルになる。	成功

4. 奏功シェア

主流：

ーゲームプレイ後コインと距離を達成するポイントをシェアすることができる。

ID	説明	期待結果	実際結果	ケース結果
23	前提条件：ユーザはゲームをします。 ーシェアボタンを押す。	ーFACEBOOK でポイントを共有する。 ー新しい近況をアップデートすることです。	ーFACEBOOK でポイントを共有する。 ー新しい近況をアップデートすることです。	成功
24	前提条件：ユーザはゲームをします。 ーシェアボタンを押す。 ーインターネット接続がない。	ーFACEBOOK でポイントを共有する。 ー新しい近況をアップデートすることができない。	ーFACEBOOK でポイントを共有する。 ー新しい近況をアップデートすることができない。	成功

5. 高得点を表す

主流：

ーマインメニュで HIGHSCORE ボタンを押して、最初からトータルの距離とトータルのコインを受けるポイントを表示して、さらに、一度で最高得点を表すポイントを表すことができる。

ID	説明	期待結果	実際結果	ケース結果
25	前提条件：ユーザはマインメニュスクリーン画面でい	ートータル距離：０ ートータルコイン：０ ー最高得距離：０	ートータル距離：０ ートータルコイン：０ ー最高得距離：０	成功

	る。 －HIGHSCORE ボタンを押す。 －最初から（初回でアプリケーションを起動する）	－最高得コイン：０	－最高得コイン：０	
26	前提条件：ユーザはマインメニュスクリーン画面でいる。 －HIGHSCORE ボタンを押す。 －最初から（初回でアプリケーションを起動して、一レースのゲームをする）。達成ポイントは距離：５５０メート、コインは１５５.	－トータル距離：５５０ －トータルコイン：１５５ －最高得距離：５５０ －最高得コイン：１５５	－トータル距離：５５０ －トータルコイン：１５５ －最高得距離：５５０ －最高得コイン：１５５	成功
27	前提条件：ユーザはマインメニュスクリーン画面でいる。 －HIGHSCORE ボタンを押す。 －達成ポイントは距離：１００メート、コインは１５.	－トータル距離：６５０ －トータルコイン：１７０ －最高得距離：５５０ －最高得コイン：１５５	－トータル距離：６５０ －トータルコイン：１７０ －最高得距離：６５０ －最高得コイン：１５５	失敗

＊結果

テストのタイプ	モード	環境テスト	ケースの数	成功の数	失敗の数
機能テスト	シングル	iPhone 5 – iOS v8.2 Samsung Galaxy S3 – Android v4.4	40	38	2
	マルチプレイヤー	Windows – Unity 4.6.1	27	26	1
単体テスト	シングル	Unity 4.6.1	98	98	0
	マルチプレイヤー	N/A	N/A	N/A	N/A
合計			165	162	3

Chapter 6 - ユーザーマニュアル

I. インストールガイド

前提条件

以下にシステム要件と重要なソフトウェアのリストである。

システム要件:

- Android
    - ▫ メモリー　: 512MB 以上
    - ▫ ハードディスク　:100MB 以上
    - ▫ OS :　Android 2.3 以上
- iOS
    - ▫ ハードウェア : IPhone, Ipad と Ipod touch
    - ▫ ハードディスク　:100MB 以上
    - ▫ OS :　iOS 7 以上

インストール手順

.apk と.ipa ファイルが CD で提供しといて、携帯電話でインストール。.apk ファイルはコピーするのは直接して, .ipa ファイルは iFunBox で iPhone でインストール。

II. ユーザーガイド

マインメニュスクリーン

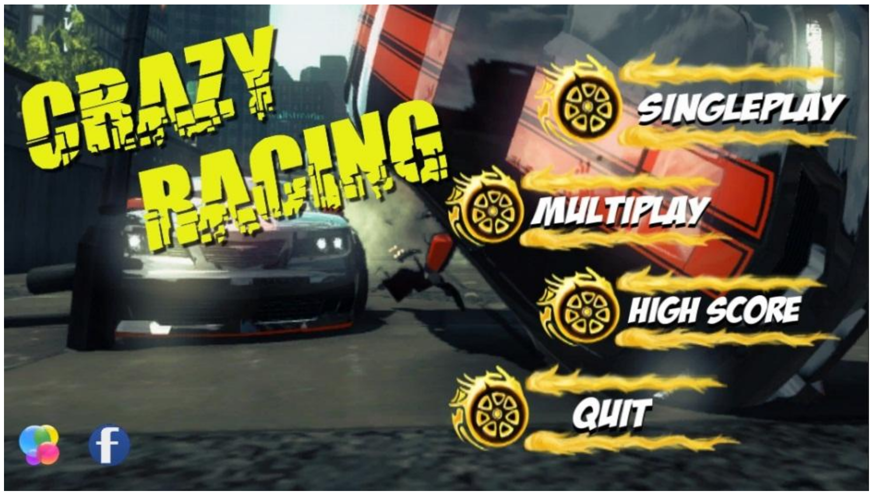


Figure 6.1: Main menu

1. ゲーム設定

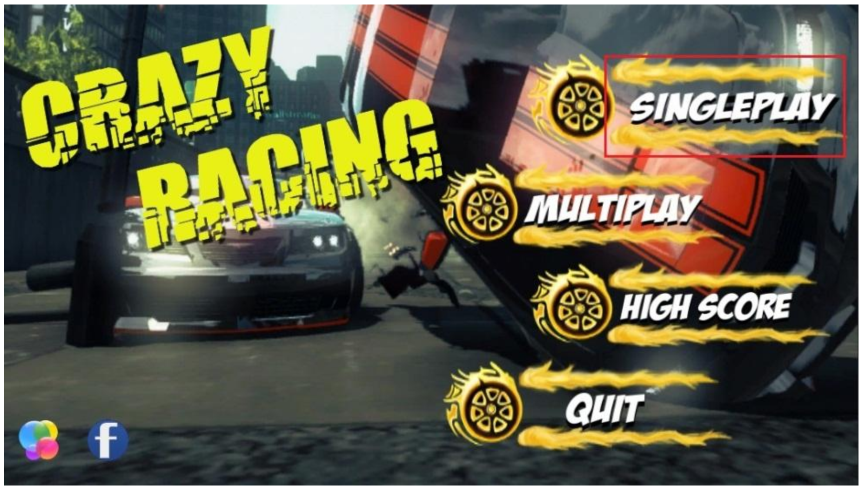


Figure 6.2: Singleplay button

マインメニュスクリーンでプレイボタンを押す

1.1 車を選んで、車をアンロック

マインメニュスクリーンでプレイボタンを押して、選んで、アンロックするスクリーンが表示する。

マインメニュスクリーンに戻るために、バックボタンを押して、ゲームをするため、次ボタンを押す。

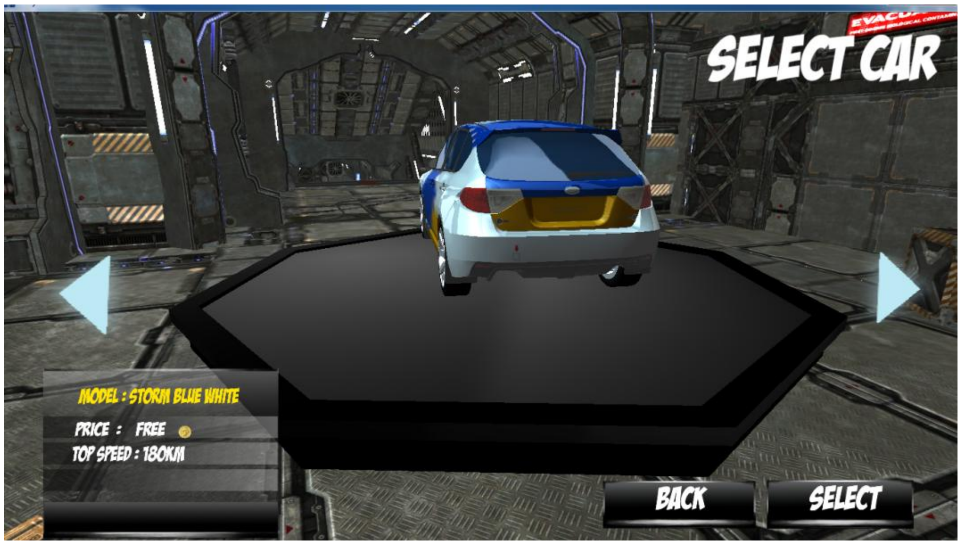


Figure 6.3: Select car

"<<"ボタンや">>"ボタンを押して、車を選ぶことができます。

新しくて、強い車を使用するため、ユーザがコインを使用して、車をアンロックして、バックボタンを選んで、マインメニュに戻る。

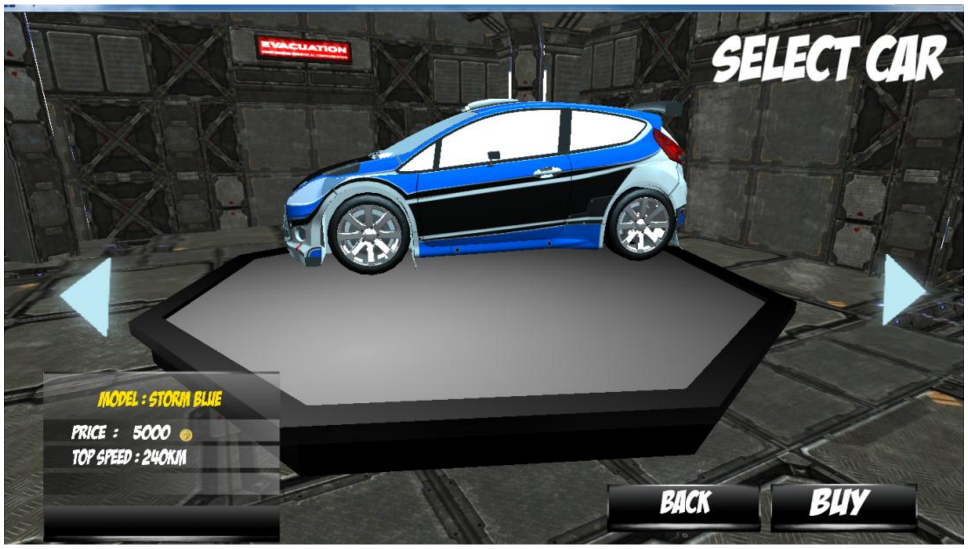


Figure 6.4: Buy car

1.2 音声を選んで、音声をアンロック

プレイヤーは車を選んでから、音声を選んで、次ボタンを押して、ゲームをする。

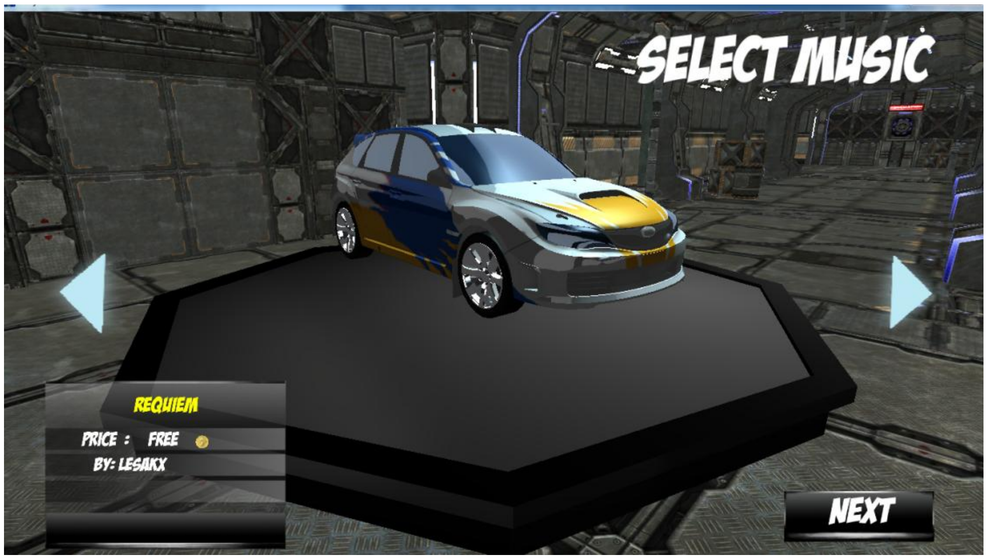


Figure 6.4: Select soundtrack

"<<"ボタンや">>"ボタンを押して、音声を選ぶことができます。

新しい音声を使用するため、ユーザがコインを使用して、音声をアンロックして、バックボタンを選んで、マインメニュに戻る。

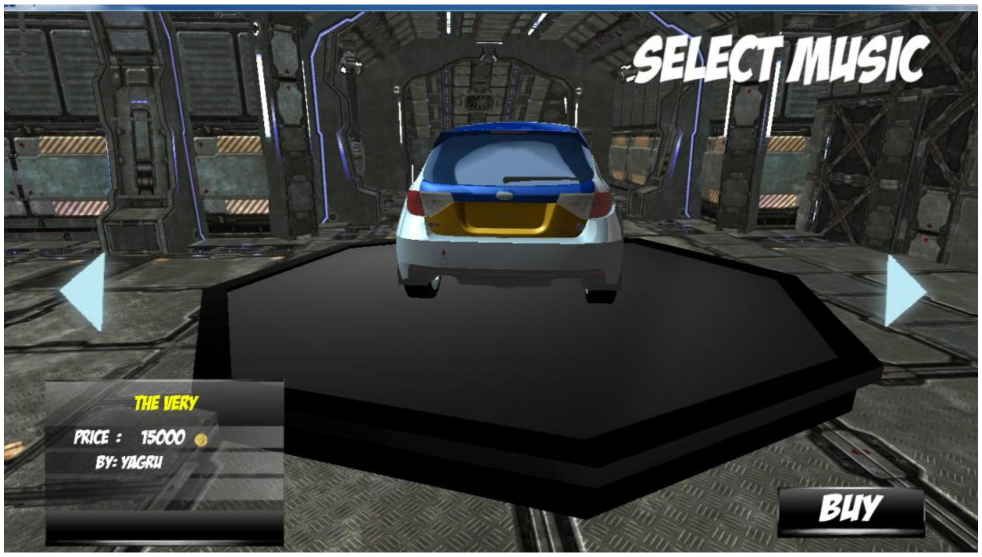


Figure 6.5: Buy soundtrack

1.3 マップ

プレイヤーは音声を選んでから、次ボタンを押して、ゲームをする。

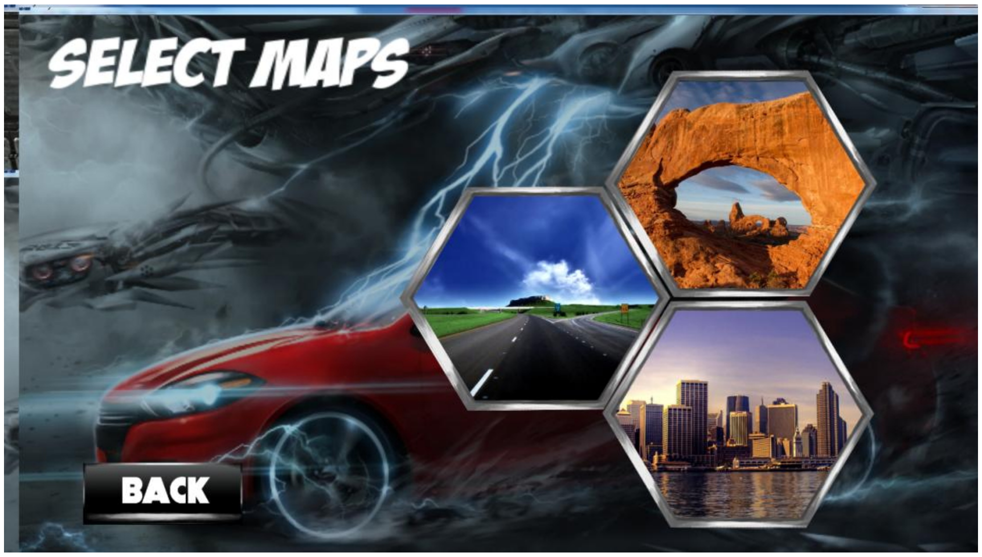


Figure 6.6: Choose map

1.4 ゲーム中

ゲーム中でユーザが携帯電話を傾けて、車を調節することができる。長い距離のためユーザが他の車を打たないように努力して、アイテムを拾して、

独特のパワーを受け取る。

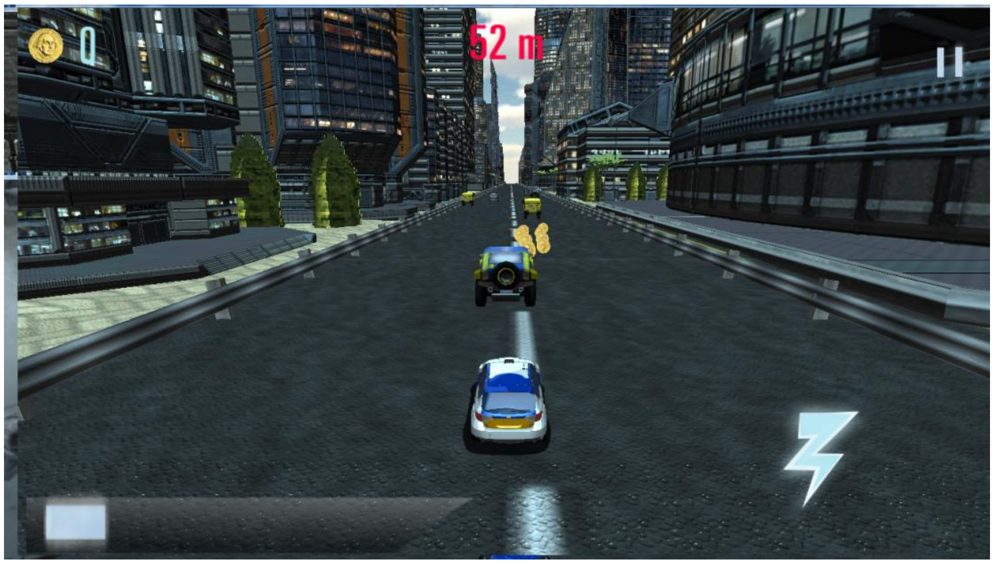


Figure 6.7: Game screen

1.4 一時停止

ボタンを押して、ゲームが一時停止

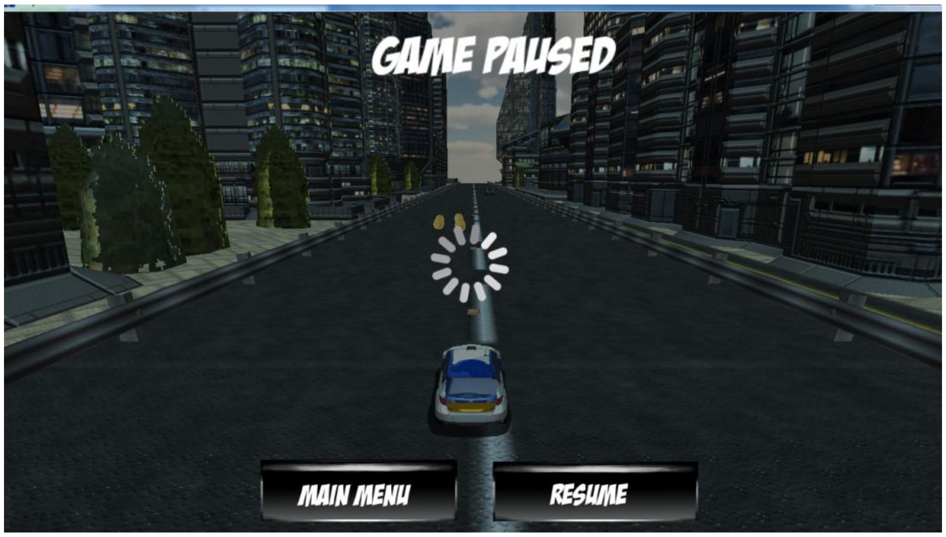


Figure 6.8: Pause game screen

次、RESUME ボタンを押して、ゲームが続けることができて、MAINMENU ボタンを押して、そのゲームを辞める。

2. ゲーム中

2.1 他の車

ユーザが他の車を打たなければばならない。

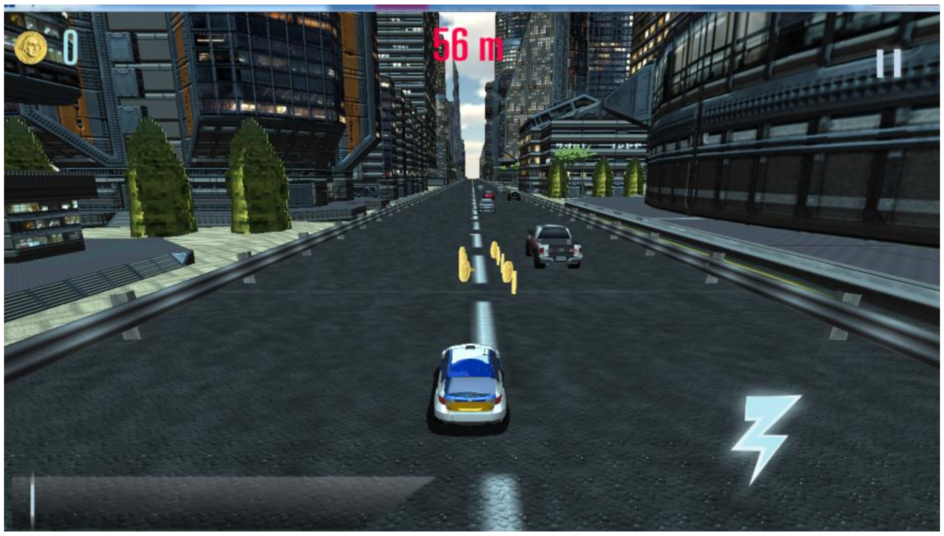


Figure 6.9: In game screen

他の車を打てば、ゲームオーバーして、コインと行った距離が表示する。

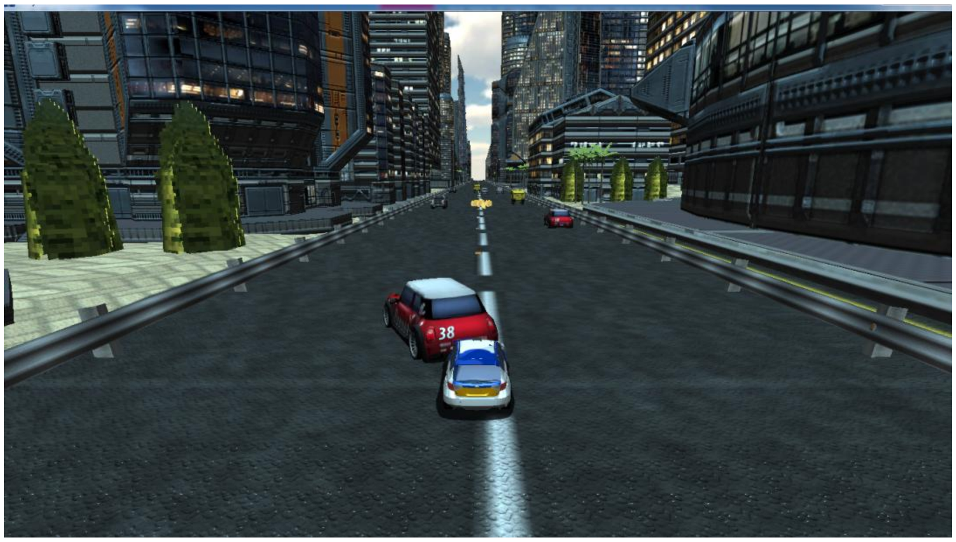


Figure 6.10: In accident screen

2.2 ヘリコプター

ゲーム中でヘリコプターが現れて、指定場所でミサイルでプレイヤーの車を攻撃する。

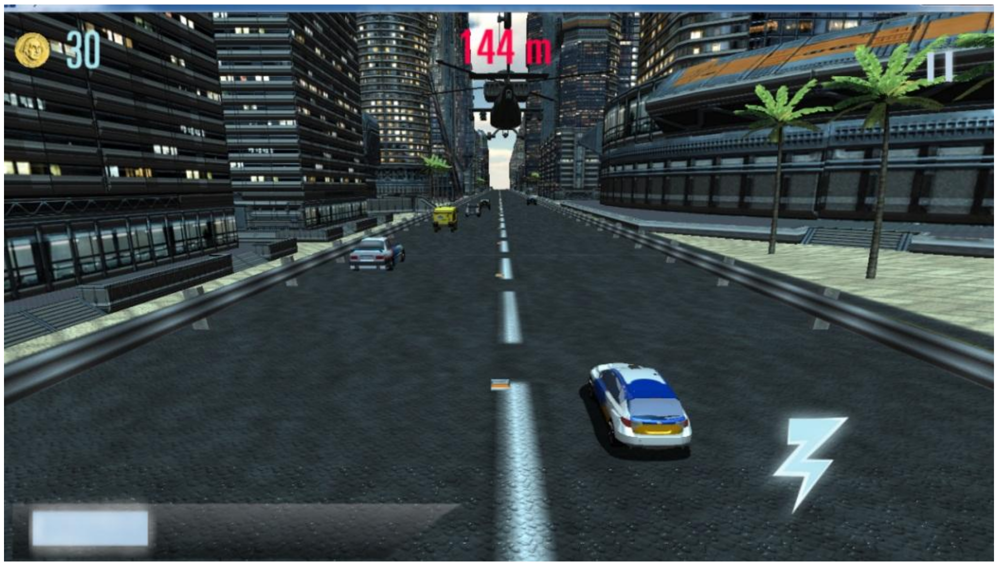


Figure 6.11: Helicopter

攻撃場所

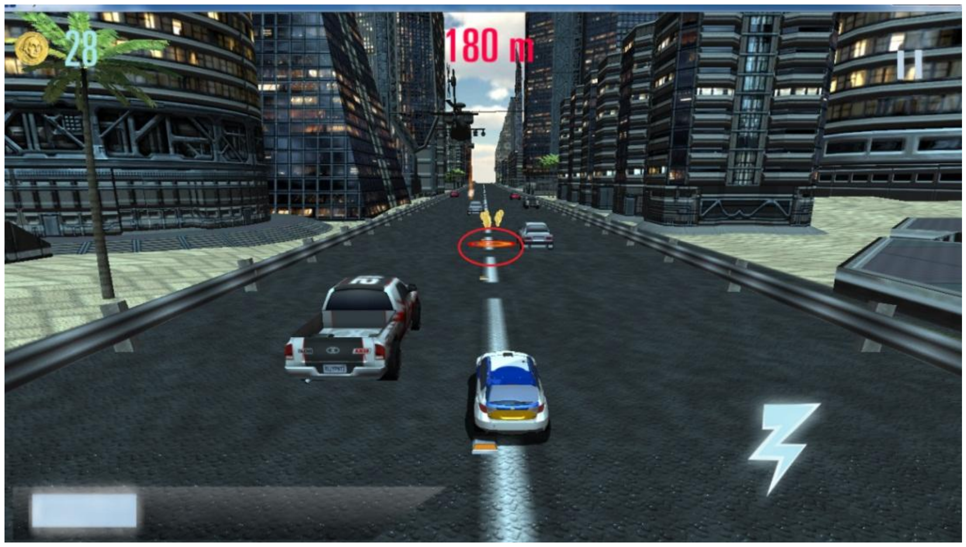


Figure 6.12: Helicopter's fight area

ミサイルを打って、車が爆発して、ゲームオーバー。

Figure 6.13: Eplosion when player's car 's going into fight area

2.3　コインと距離

コインを収集するために、ユーザが車を調節して、スピドブーストすることができるため、コインを収集すれば、収集して、エネルギーが上がる。

距離が長くになれば、長いほど、速い速度で運動する。

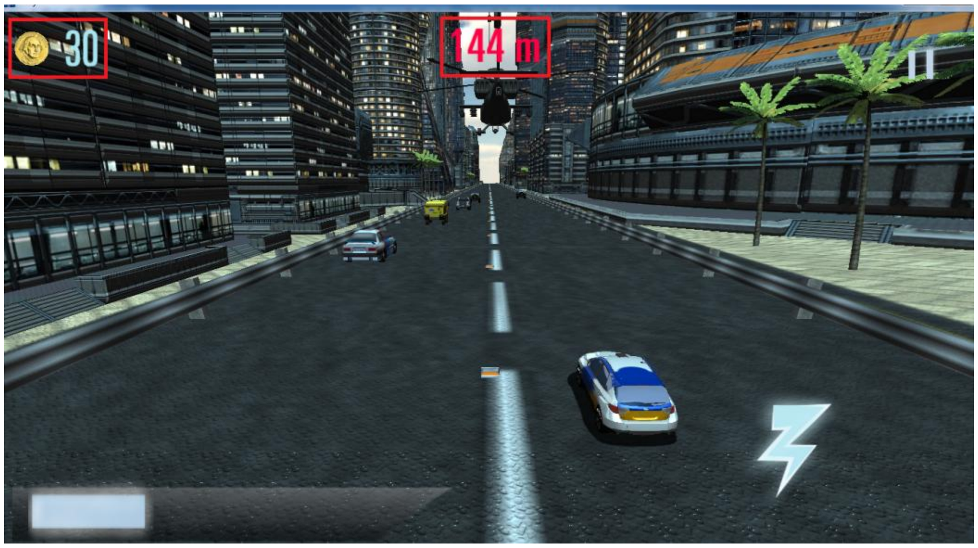


Figure 6.14: Coins and distance index

2.4 エネルギーとスピードブースト

左側、下でエネルギーバールがあって、右側、下でスピードボタンがある。

コインを収集するために、ユーザが車を調節して、スピドブーストすることができるため、コインを収集すれば、収集して、エネルギーが上がる。

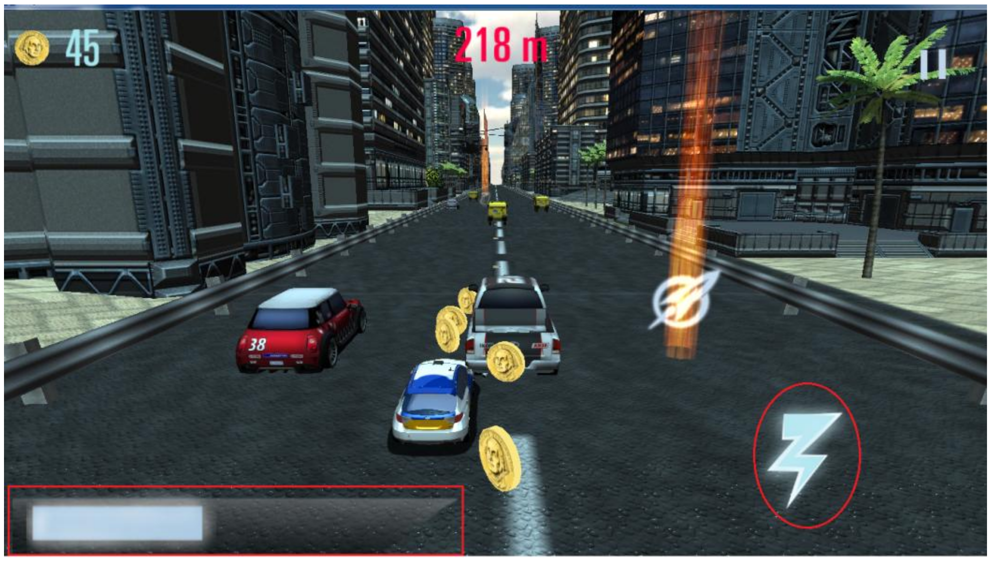


Figure 6.15: Nitrous bar and accelerate button

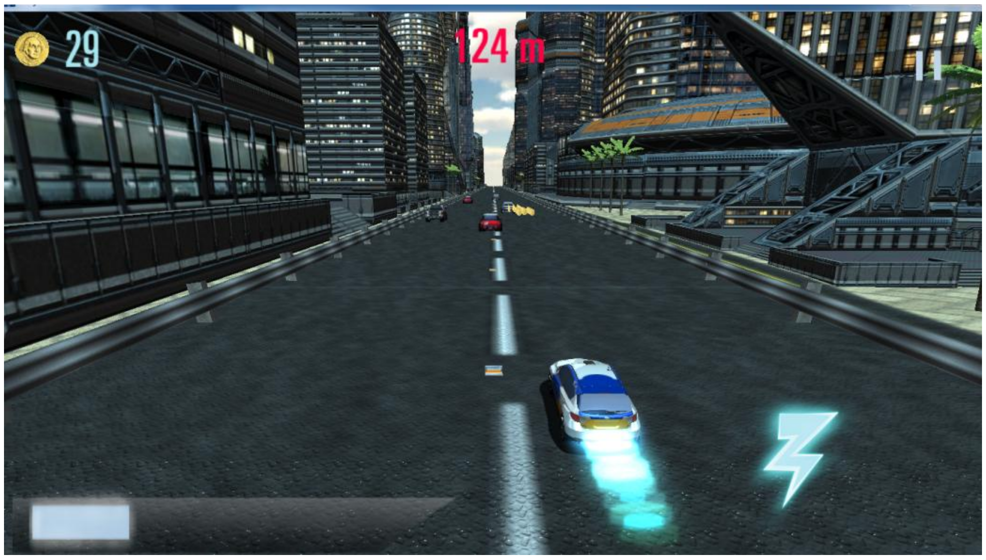


Figure 6.16: In acceleration

ブーストスピード中

2.5 アイテム

2.5.1 コインを二倍に受ける

７秒以内、コインを二倍受け取ることができる。

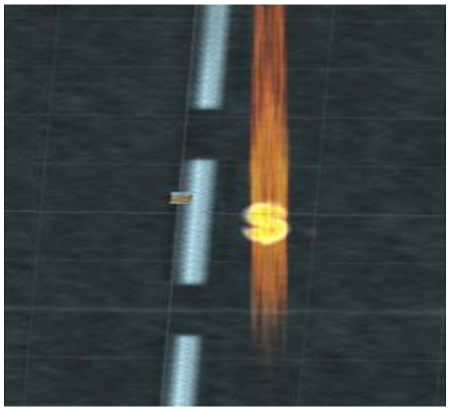


Figure 6.16: "Double coin" Power-up

2.5.2 スルーに運動

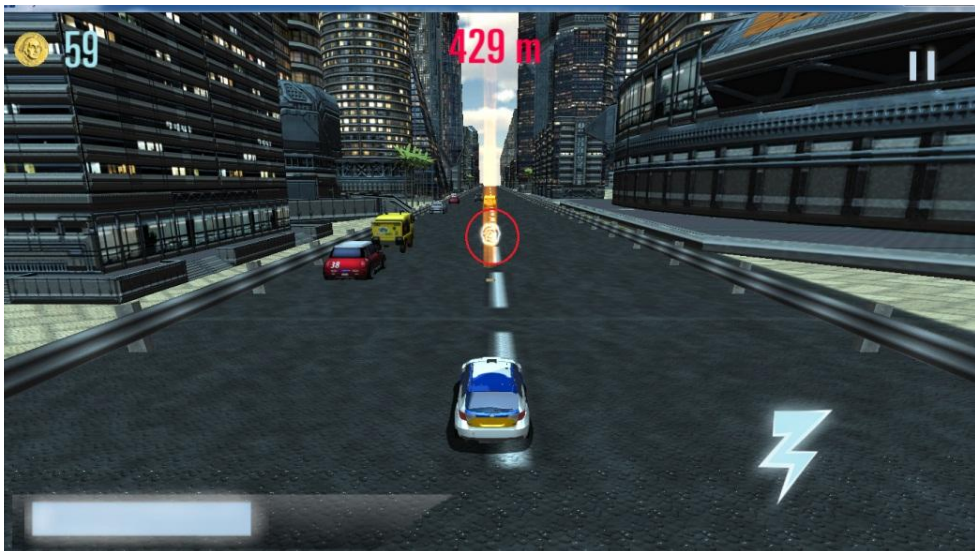


Figure 6.17: "Ghost" Power-up

他の車を打たないことができる。

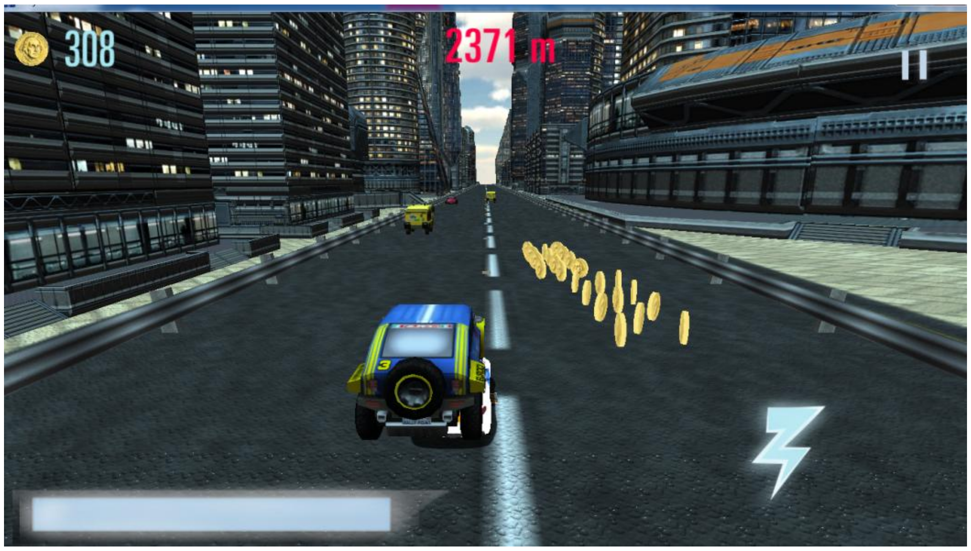


Figure 6.18: Going through effect

2.5.3 瞬時エネルギー

エネルギーバールがすぐフールになる

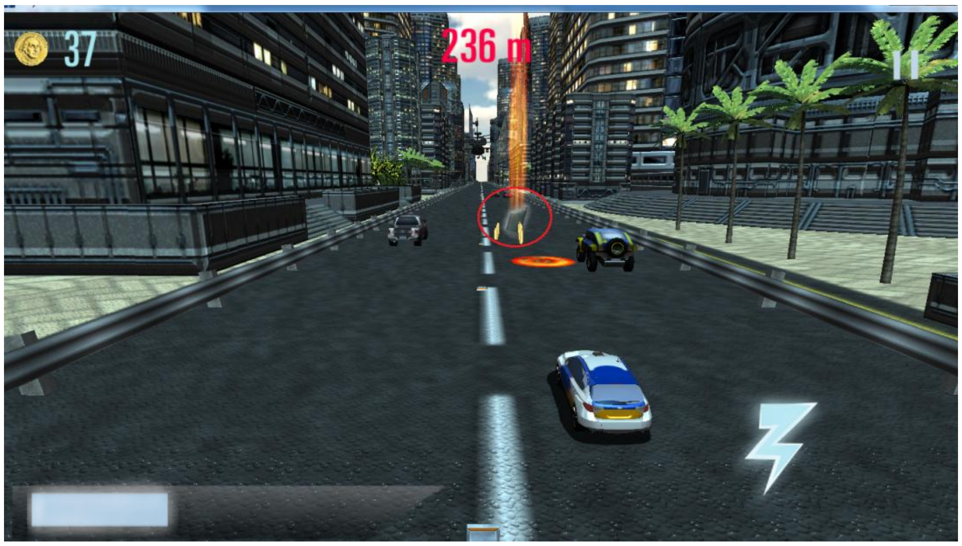


Figure 6.19: “Nitrous” Power-up

2.5.4 磁石

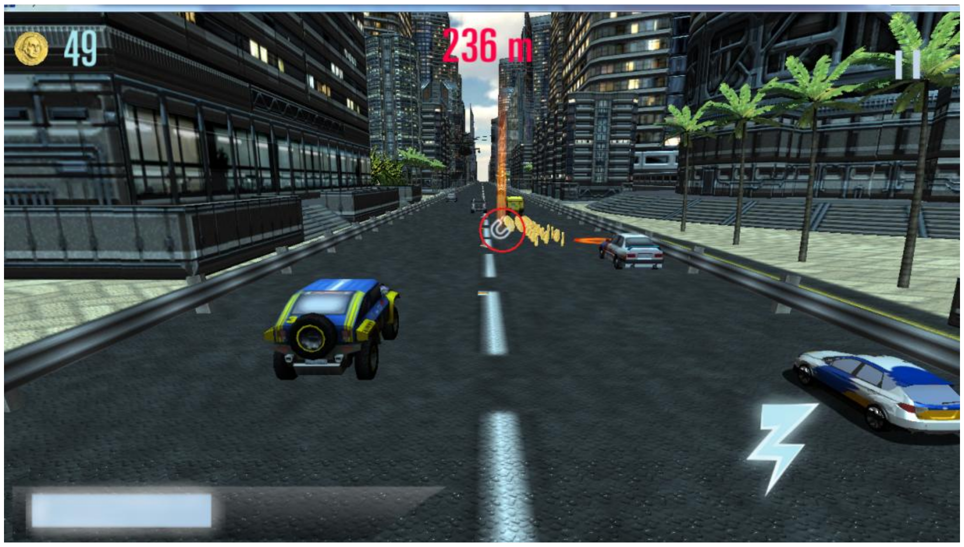


Figure 6.20: "Magnetic" Power-up

自動的で周りコインを収集することです。

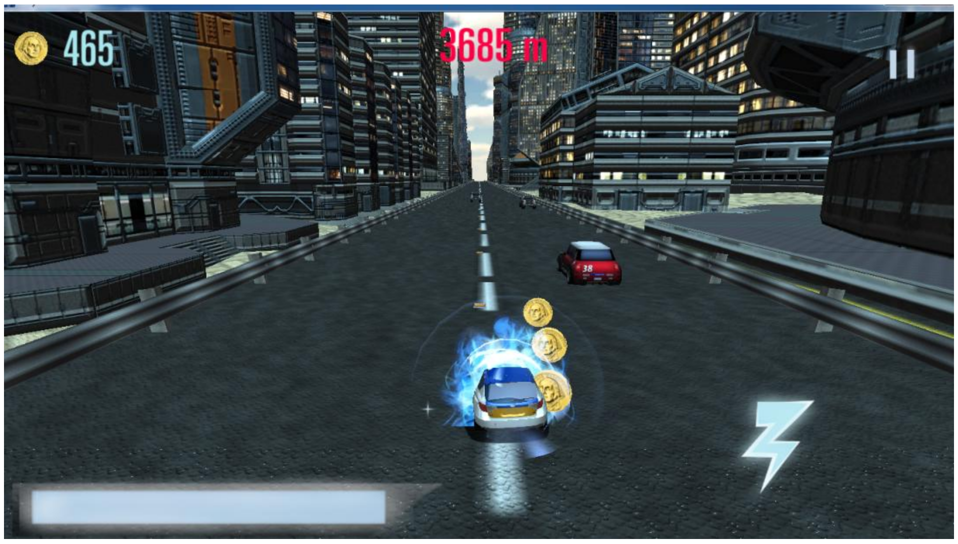


Figure 6.21: In "Magnetic" effect

2.5.5 バズーカ

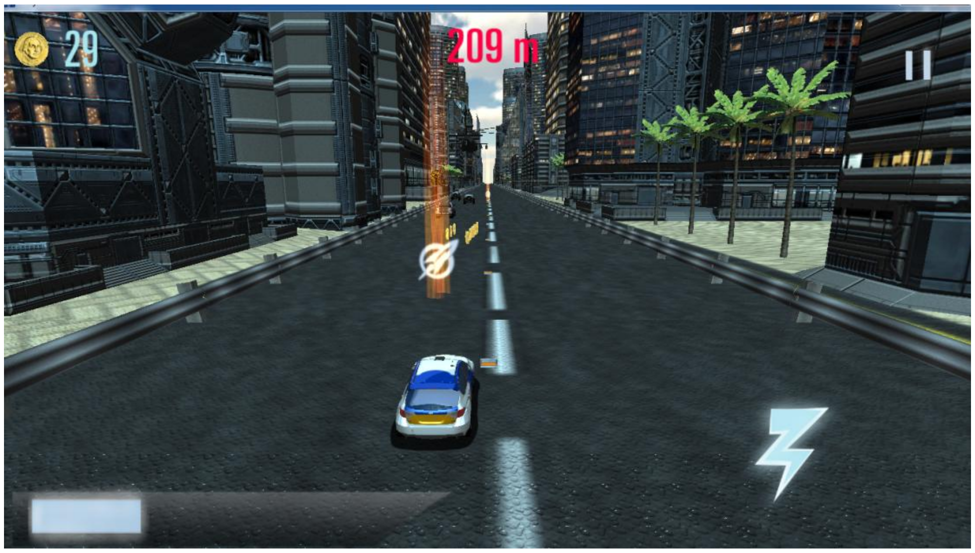


Figure 6.22: Bazooka

自動的にヘリコプターを消滅することができる。

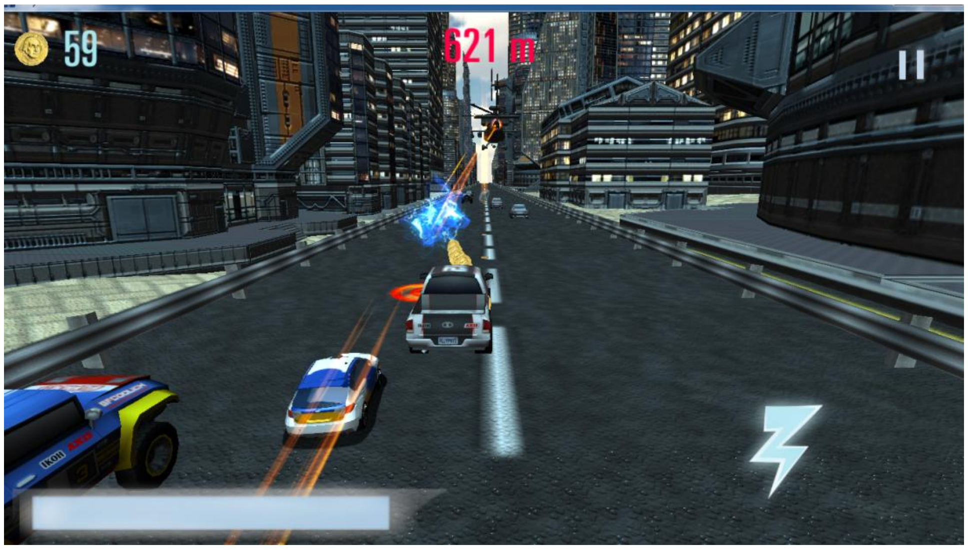


Figure 6.23: Fighting back

2.6 ゲーム後

ゲームオーバー後、コインと行った距離が表される。

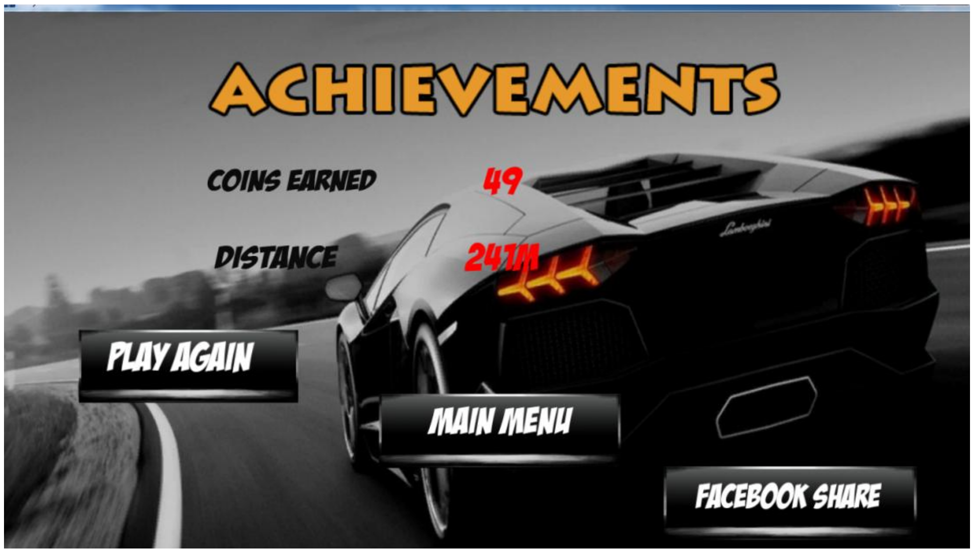


Figure 6.24: Achievement board

FACEBOOK で得点がシェアすることができる。

3. ハイスコア

HIGHSCORE ボタンを押して、ハイスコアを表示することができる。

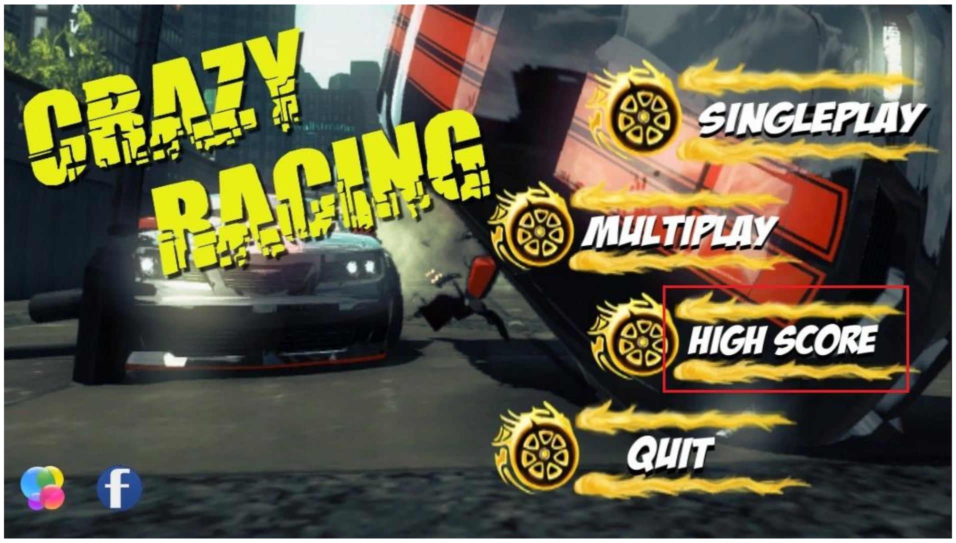


Figure 6.25: High score button

ハイスコアスクリーン。

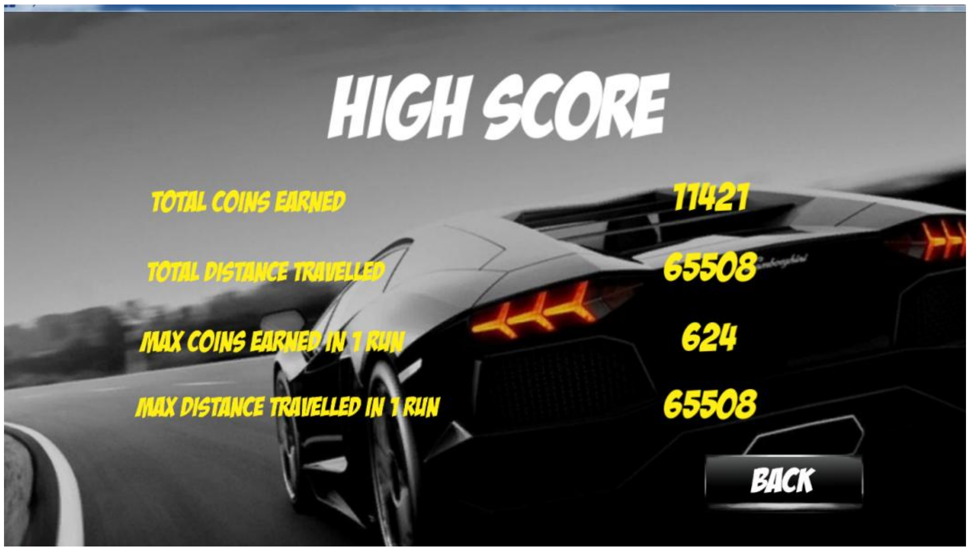


Figure 6.26: High score board

Total coin earned や Total distance travelled は最初からトータルの収集したコインや行った距離が現れる。Max coins earned in 1 run や Max distance travelled in 1 run　は 1 レースでベスト結果です。

4. マルチプレイヤー

マインメニュスクリーンで MultiPlay ボタンを押して、

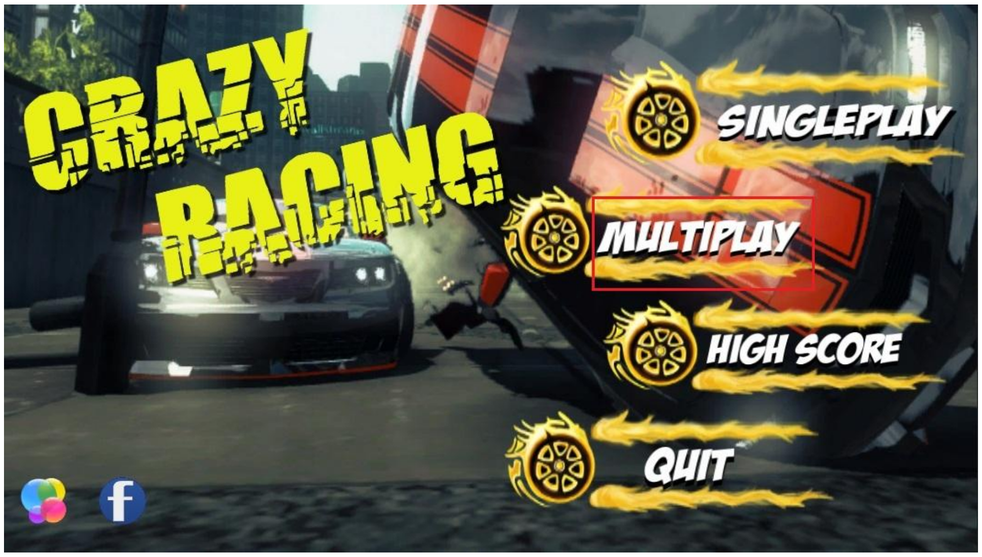


Figure 6.27: Multiplay button

マルチプレイヤーモードで、一人プレイヤーがホストの役割を担当して、他のプレイヤーがクライアントの役割を担当する。

指定距離を達する時とか、他の車が壊れる時、勝手が決定することができます。

5. ゲームを辞める

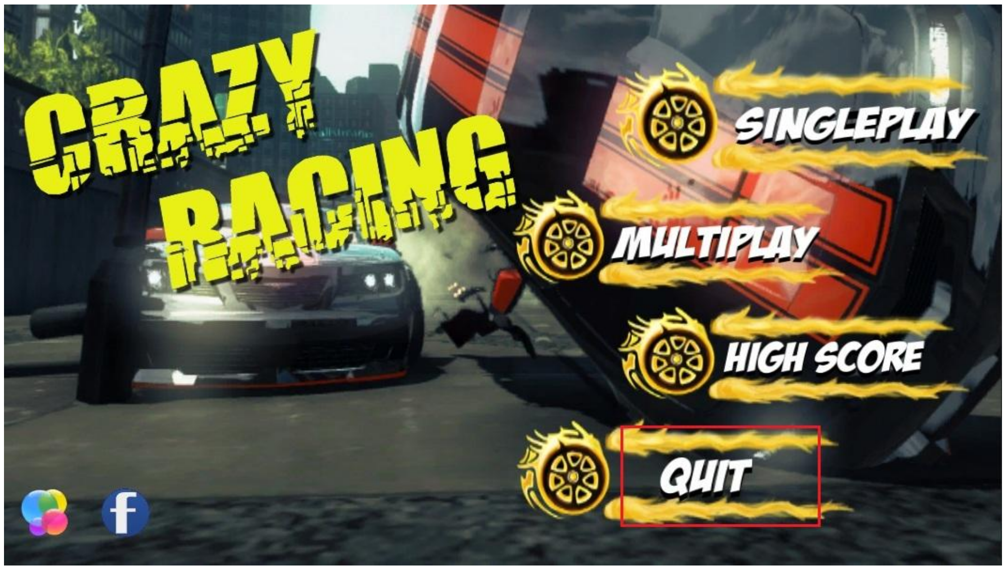


Figure 6.28: Quit button

ゲームを出て、QUIT ボタンを押す。